UNIVERSIDAD DE ZARAGOZA

CENTRO POLITÉCNICO SUPERIOR

PROYECTO FIN DE CARRERA
Ingeniería de Telecomunicación

Implementación de una plataforma de telemonitorización de pacientes estándar, *open source* y ubicua basada en ISO/IEEE 11073-PHD sobre Android.

Julio, 2010

Pedro Funes Salas

Director
Pilar del Valle García

Ponente
Ignacio Martínez Ruiz

GRUPO DE TECNOLOGÍAS DE LAS COMUNICACIONES

Resumen

La telemedicina se entiende como el uso de las tecnologías de la información y las telecomunicaciones para el diagnóstico médico y cuidado de pacientes, ofreciendo un servicio cómodo y mejorando la calidad de vida de los usuarios.

En ese ámbito se enmarca este PFC, que continúa con la línea de investigación llevada a cabo en los últimos años por el grupo X73Spain de la Universidad de Zaragoza, abriendo la plataforma de telemonitorización existente a nuevos entornos *open source* basados en el sistema operativo Android, y añadiendo soporte para nuevos dispositivos médicos.

El *software* se ha desarrollado en Java, más concretamente en J2ME (*Java Mobile Edition*), por tratarse del lenguaje indicado para este sistema operativo, y ser además un entorno libre de trabajo (*open source*).

Android ofrece un gran potencial para la implementación de nuevas funcionalidades y servicios. Al tratarse de un entorno libre, permite una rápida integración con otras plataformas, además de ofrecer una serie de herramientas que facilitan el desarrollo de aplicaciones. Por otro lado, los terminales existentes que cuentan con este sistema operativo, poseen unas prestaciones de muy alto nivel: pantallas de gran tamaño y calidad visual, cámaras con alta definición, procesadores eficientes, etc. Todas estas características hacen de Android un entorno más que apropiado para la elaboración de este proyecto.

Para conseguir la interoperabilidad de esta plataforma con otros entornos de e-Salud existentes, es necesaria la aplicación de un estándar que proponga una serie de normas en cuanto a intercambio de datos, representación de la información y gestión de dispositivos médicos. Se establece entonces la norma ISO/IEEE 11073 (X73) como guía en el desarrollo de este PFC.

Este proyecto proporciona un entorno ubicuo donde se facilita el intercambio de información médica entre los diferentes dispositivos médicos que pueda necesitar el paciente, y en diversos casos de uso. De esta forma, se dispone de un terminal Agente, que representa varios dispositivos médicos estándar (termómetro, tensiómetro, pulsioxímetro y báscula), y otro terminal haciendo las funciones de Manager, ambos simulados en teléfonos Android.

Me gustaría agradecer especialmente a Pilar, a Nacho y al resto de compañeros, su disponibilidad y ayuda prestada en todo momento.

Y como no, a las personas cercanas, mi familia y amigos, por su paciencia y comprensión, pero sobre todo, por estar ahí.

"Ninguna fuerza doma, ningún tiempo consume, ningún mérito iguala, el nombre de la libertad."

N. Machiavelli

Índice de contenidos

Índice de figuras

Índice de tablas

Acrónimos y siglas

AENOR	Asociación Española de NORmalización
ACR	Colegio Americano de Radiólogos
ANSI	American National Standards Institute
API	Application Programming Interface
ASL2	Apache Software License 2
ASN1	Abstract Syntax Notation One
AVD	Android Virtual Device
BER	Basic Encoding Rules
CE	Compute Engine
CEN	Comité Europeo de Normalización
DER	Distinguished Encoding Rules
DIM	Domain Information Model
ECG	ElectroCardioGrama
EHR	Electronic Healthcare Record
EDR	Enhanced Data Rate
FSM	Finite State Machine
GPRS	General Packet Radio Service
GUI	Graphic User Interface
HCE	Historia Clínica Electrónica
HDP	Health Device Profile
HL7	Health Level 7
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IHE	Integrating the Healthcare Enterprise
IP	Internet Protocol
IrDA	Infrared Data Association
ISCIII	Instituto de Salud Carlos III
ISO	International Organization for Standardization
J2ME	Java 2 Mobile Edition
MCAP	Multi-Channel Adaptation Protocol
MD	Medical Device
MDS	Medical Device System
MIT	Massachusetts Institute of Technology
NEMA	Asociación Americana de Fabricantes Eléctricos
OLPC	One Laptop Per Child
ONG	Organización No Gubernamental
OSI	Open Systems Interconnection
PC	Personal Computer
PER	Packed Encoding Rules
PFC	Proyecto Fin de Carrera
PHD	Personal Health Device
PHDC	Personal Health Device Class
PoC	Point of Care
RFCOMM	Radio Frequency Communication
SCP – ECG	Standard Communications Protocol for computer assisted Electrocardiography
SPP	Serial Port Profile
SDK	Software Development Kit
SO	Sistema Operativo
TCP	Transmission Control Protocol
USB	Universal Serial Bus
WiFi	Wireless Fidelity
XML	Extensible Markup Language
UCI	Unidad de Cuidados Intensivos
UPNA	Universidad Politécnica de Navarra
URJC	Universidad Rey Juan Carlos
UZ	Universidad de Zaragoza

1. Introducción

En esta sección se introduce el concepto de telemedicina y se justifica la necesidad de usar un estándar de interoperabilidad, la norma ISO/IEEE 11073, entre dispositivos médicos. También se hace una pequeña referencia al sistema operativo utilizado, Android, y a los antecedentes existentes relacionados con este proyecto. Finalmente se definen los objetivos y la estructura de la memoria.

1.1 Telemedicina y necesidad de cubrir nuevos estándares

El término telemedicina [1]-[2] se puede definir, en un sentido amplio, como el uso de las tecnologías de la información y las telecomunicaciones para proporcionar servicios sanitarios, formación e información a profesionales de la salud y consumidores. Es decir, se trata de la utilización de la tecnología para el diagnóstico médico y el cuidado de pacientes.

La evolución más reciente de las nuevas tecnologías, y de las herramientas que proporciona la nueva Sociedad de la Información, permite aplicar este concepto de telemedicina o e-Salud a innumerables campos: aplicaciones asistenciales (teleconsulta, telediagnóstico, telemonitorización), administración y gestión de pacientes, o formación e información tanto de profesionales como usuarios [3].

Debido a la relativa novedad de este sector, existe una carencia importante de estandarización de dispositivos médicos, tanto en la toma de medidas, como en la transmisión de éstas. Esto conlleva, por un lado, un importante aumento de los costes, ya que para cada marca y dispositivo, se tendría que utilizar un *software* propietario y, por otro lado y más importante todavía, una elevada incomodidad tanto para el personal médico como para los pacientes, que se verían obligados a comprar todos los equipos de la misma marca o, en caso de no hacerlo, aprender a manejar diferentes dispositivos, cada uno con un comportamiento particular establecido por el fabricante [4].

La mayoría de sistemas de telemonitorización que se han desarrollado en los últimos años arrastran una cierta heterogeneidad que no los hacen integrables con otras aplicaciones similares y en el contexto sanitario global. Para ello, en la actualidad, un aspecto clave a tener en cuenta es el diseño basado en normas que definan completamente la forma en la que los dispositivos médicos se conectan al sistema, encapsulan los datos y los transmiten; es decir, un protocolo estándar [5].

1.2 Norma ISO / IEEE 11073

El estándar para la comunicación de dispositivos médicos asumido dentro de la Unión Europea es la norma ISO/IEEE 11073 (X73). Esta familia de estándares abarca los siete niveles de la pila de protocolos OSI (ver Figura 1.1) y proporciona la versatilidad suficiente para convertir la información en un formato interoperable de manera que pueda ser intercambiada entre un dispositivo de salud personal y un sistema central de registro. Los datos posteriormente pueden ser enviados a un centro remoto de control o almacenamiento de Historia Clínica Electrónica (HCE), así se permite construir sistemas más completos que constituyan soluciones globales.

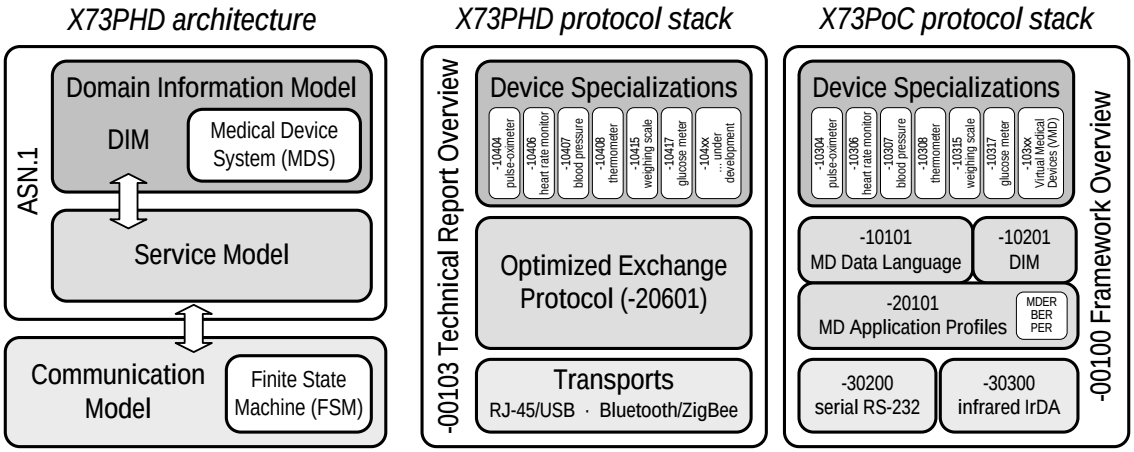


Figura 1.1 – Arquitectura de la norma IEEE 11073 y evolución de X73PoC a X73PHD

La norma X73, que nació centrada en el Punto de Cuidado (*Point of Care*, X73-PoC) [6] del paciente y enfocada a su implantación en Unidades de Cuidados Intensivos (UCIs), ha experimentado en los últimos años diversas evoluciones no contempladas inicialmente (nuevos casos de uso, nuevas tecnologías de interconexión, nuevos perfiles, etc.). En esta línea, el Comité Europeo de Normalización (CEN) fue capaz de adaptarse a esta realidad tecnológica ofreciendo una nueva versión de X73, manteniendo intactos los aspectos más

característicos y optimizando otros acorde a las nuevas tendencias: el denominado *Personal Health Device* (X73-PHD) [7]. (Para más información sobre el estándar, ver Anexo A).

1.3 Android

Android [8] es un Sistema Operativo (SO) para dispositivos móviles basado en el núcleo Linux. Inicialmente fue desarrollado por Android Inc., compañía que fue comprada después por Google. En la actualidad lo desarrollan los miembros de la Open Handset Alliance [9], un consorcio de compañías de hardware, software y telecomunicaciones comprometidas con la promoción de estándares abiertos para dispositivos móviles, liderado por Google. Su lanzamiento se produjo en Noviembre de 2007 con la versión 1.1, y ha evolucionado a una velocidad vertiginosa hasta llegar a la versión actual 2.1 (Flan), pasando antes por la 1.5 (Cupcake), 1.6 (Donut), y 2.0 (Eclair).

Es un paquete que engloba un SO, un *runtime* de ejecución basado en Java, un conjunto de librerías de bajo y medio nivel y un repertorio inicial de aplicaciones destinadas al usuario final. Se distribuye bajo una licencia Apache Versión 2 (ASL2), una licencia libre permisiva que permite la integración con soluciones de código propietario. Este desarrollo de aplicaciones por terceros se lleva a cabo en lenguaje Java y a través del *Software Development Kit* (SDK) proporcionado por el mismo Google. (Más información en el Anexo B).

1.4 Antecedentes del proyecto

Este PFC es la evolución de una plataforma desarrollada por el grupo X73Spain [10], formado por la Universidad de Zaragoza (UZ), la Universidad Politécnica de Navarra (UPNA) y el Instituto de Salud Carlos III (ISCIII), que se ha ido adaptando a las evoluciones sufridas por la norma X73 (ver Figura 1.2).

Figura 1.2 – Plataforma de telemonitorización extremo a extremo

La primera versión de la aplicación de la norma, plataforma 1.0-alfa, estaba basada en X73-PoC y dotaba de interoperabilidad y configuración *plug&play* a dispositivos médicos asignados a la monitorización de pacientes.

Posteriormente se desarrolló la plataforma 1.5-beta, que es una arquitectura de migración, actualizada a la norma X73-PHD. El principal motivo para la implementación de esta actualización de la plataforma es la complejidad en el desarrollo con la plataforma 1.0-alfa (X73-PoC). Esto se debe al hecho de haber diseñado un código basado en librerías orientadas a Linux y disponer de una elevada cantidad de llamadas a funciones para representar el total de las especificaciones en cada uno de los niveles OSI que define la norma. La solución es reutilizar la plataforma encapsulándola para poder hacerla funcionar en un entorno Windows y simplificar el código para facilitar el trabajo sobre el mismo. Se implementa la parte de la comunicación entre un dispositivo médico (MD, *Medical Device*) y un PC actuando de *Gateway* (CE, *Compute Engine*).

La segunda versión basada plenamente en X73-PHD, plataforma 2.0-release, se encarga de solucionar los problemas de la anterior versión. Se orienta a entornos ubicuos y dispositivos llevables y se abre a nuevas funcionalidades *plug&play* y de gestión remota; se realiza la migración a esta plataforma para adaptarla a la nueva arquitectura requerida para dichas funcionalidades y que posibilite el desarrollo de sistemas *end-to-end* basados en estándares. Se implementa el protocolo de comunicación entre MD-CE, simulados ambos por PCs, y cumpliendo la norma X73-PHD.

Por último se desarrolló la plataforma 2.1-BT, que mantenía la estructura de la pila y los subsistemas de las anteriores plataformas, pero se modificaba el *software* heredado para adaptarlo a dispositivos móviles comunicándose a través de tecnología Bluetooth, sobre un SO propietario como Windows Mobile [11].

Por otro lado se encuentra el trabajo realizado por el grupo GSyC/LibreSoft de la Universidad Rey Juan Carlos de Madrid (URJC), a través de su proyecto OpenHealth [12]. Inicialmente se trataba de una implementación de la norma IEEE 11073-20601 para la conexión de un agente termómetro (simulado) y un manager, basados ambos en Android, a través de Bluetooth (RFCOMM). Sin embargo, su trabajo ha evolucionado rápidamente y actualmente han conseguido conectar un pulsioxímetro real con un manager Android (NexusOne) a través de Bluetooth, implementando para ello la pila HDP (*Health Device Profile*) + MCAP (*Multi-Channel Adaptation Protocol*).

Tanto el proyecto OpenHealth como la plataforma 2.1-BT mencionada anteriormente, constituyen la base de este PFC.

1.5 Motivación

La idea de este proyecto nace basada en tres motivos fundamentales. Primero, debido al afán por llegar a un mayor número de usuarios y entornos, a través de diferentes plataformas, por lo que es necesaria la migración a un sistema de código abierto (*open source*) como Android. Segundo, se persigue la optimización de la norma X73 para dispositivos de salud personal (PHD). Y tercero, para implementar nuevas especializaciones de dispositivos médicos.

Android es un sistema operativo multitarea especialmente apropiado para ampliar el abanico de posibilidades en cuanto a diversidad de casos de uso se refiere, ya que ofrece un entorno de desarrollo avanzado, y los terminales que lo implementan cuentan con unas prestaciones de alto nivel. Y al tratarse de *software* libre, permite una fácil integración con otras implementaciones y plataformas, algo que no ocurre con los sistemas propietarios.

Dentro de la convulsión existente hoy en día en cuanto a sistemas operativos y nuevos dispositivos como los recientes *tablets*, parece ser que Android se encuentra un paso por delante en lo que a homogeneidad se refiere. La mayor parte de fabricantes están apostando por él, y esto hace que la elección de migrar la plataforma de telemonitorización a este nuevo SO sea la más adecuada.

Además de los aspectos técnicos, existe también la motivación de realizar un proyecto real que, en un momento dado, pueda llevarse a la práctica, y qué mejor forma de hacerlo que desarrollando una plataforma de e-Salud que ayude al seguimiento domiciliario de pacientes, haciendo más confortable la vida de éstos y mejorando la calidad de la atención médica. Con la utilización de este sistema, ni el paciente debería ir al hospital para tomarse medidas rutinarias como la temperatura, la tensión o el peso, ni los servicios médicos tendrían que desplazarse a casa del paciente, ahorrándose de esta forma tanto tiempo como dinero. Si a todas estas comodidades se añade la utilización de una tecnología sin cables como Bluetooth, las facilidades ofrecidas son extraordinarias.

1.6 Objetivos

El objetivo principal de este PFC es la implantación de la norma X73-PHD en una nueva plataforma *open source* como Android. Para ello se partirá de la plataforma 2.1-BT desarrollada sobre Windows Mobile, y se procederá a la migración del *framework* de desarrollo para su implementación en dispositivos móviles HTC G2 (*Google Dev Phone*) [13]. Por lo tanto se deberá adaptar el diseño de la máquina de estados de la norma X73-PHD a la nueva plataforma, atendiendo fundamentalmente a principios de abstracción de *software*, modularidad y escalabilidad.

Esta nueva solución incorporará además nuevos dispositivos médicos simulados sobre los *Google Dev Phone* mencionados y nuevos interfaces de usuario personalizados para los escenarios de salud personal. Concretamente se implementará el termómetro (IEEE 11073-10408), el tensiómetro (IEEE 11073-10407), el pulsioxímetro (IEEE 11073-10404) y la báscula (IEEE 11073-10415).

Al tratarse de un entorno *open source* se podrá integrar de forma inmediata con otras implementaciones libres, como por ejemplo la pila de Bluetooth HDP, que no están disponibles para otros sistemas como Windows. Además, permitirá la interoperabilidad con dispositivos médicos comerciales certificados por Continua, y la integración total con el resto de elementos que forman la plataforma completa de telemonitorización *end-to-end.*

Debido a las franjas y tarifas de conexión existentes en la actualidad, se ofrecerá la posibilidad de guardar las medidas realizadas en ficheros XML locales, lo que permitirá tanto su posterior consulta, como el envío de éstas a un servidor de HCE.

1.7 Estructura de la memoria

A continuación se muestra cómo está estructurada la memoria y qué contiene cada una de las secciones de las que se compone:

- **2. ESTADO DEL ARTE:** En este capítulo se van a presentar el estado de los organismos y normas para e-Salud que existen actualmente, así como los proyectos existentes en los que aparece el SO Android enfocado a sistemas de salud. Se muestran los primeros dispositivos médicos que cumplen con la norma X73, certificados por Continua Alliance, así como las últimas novedades en este campo.

- **3. ANÁLISIS Y DISEÑO:** Se analizan las necesidades de los usuarios/pacientes que, en diferentes casos de uso, deberán tomarse una serie de medidas biomédicas; se plantea el diseño y arquitectura adoptados, siguiendo las especificaciones de la norma X73.

- **4. DESARROLLO E IMPLEMENTACIÓN:** Se expone el *software* desarrollado con explicaciones de bajo nivel, detallando partes importantes de la programación; se explica el funcionamiento de la plataforma a nivel de usuario.

- **5. EVALUACIÓN Y RESULTADOS:** Como última parte de la fase de desarrollo, se muestran la serie de pruebas llevadas a cabo para comprobar que la implementación realiza correctamente las tareas para las cuales ha sido diseñada.

- **6. CRONOGRAMA DE IMPLANTACIÓN:** Este capítulo sirve para explicar cuál ha sido el cronograma de implantación del PFC, y cómo se han desarrollado las diferentes fases de las que se compone dicho proyecto; además se hará una representación del diagrama de Gantt para reflejar el transcurso de este proyecto.

- **7. CONCLUSIONES Y LÍNEAS FUTURAS:** En este apartado se expone qué se ha hecho y se establecen las conclusiones extraídas del proyecto. Se explican también las líneas futuras que podrían seguirse tras este proyecto, así como las posibles mejoras que se podrían añadir.

- **ANEXO A. La norma ISO/IEEE 11073:** Breve descripción del estándar.

- **ANEXO B. Android:** Introducción al sistema operativo Android y al desarrollo de aplicaciones para esta plataforma con Eclipse. Comparativa de terminales basados en Android.

- **ANEXO C. Definición de medidas:** Definición de la temperatura, presión arterial, ritmo cardiaco, nivel de oxígeno en sangre y peso, y descripción de cómo realizar estas medidas correctamente.

2. Estado del arte

En este apartado se presentan los distintos organismos de e-Salud que existen actualmente, así como las normas emitidas por éstos. Se muestran además diferentes modelos de dispositivos médicos utilizados en este campo, viéndose la escasa presencia de aquellos que cumplen con la norma X73. En esta dirección trabaja la Continua Alliance, la cual ha presentado varios dispositivos que sí cumplen este estándar. Y finalmente se explicará la evolución sufrida por la plataforma de telemonitorización desarrollada por este grupo.

2.1 Organismos y normas para la salud

Las principales organizaciones encargadas de la Informática Médica y las TICs para la salud y que, entre otras actividades importantes, colaboran en el desarrollo de estándares y normas, objeto de estudio en este trabajo, son:

- ✓ **CEN, Comité Europeo de Normalización**, a través de su Comité Técnico CEN/TC251, es la principal organización europea con competencia en este campo.
- ✓ **AENOR, Asociación Española de NORmalización**, mediante su Comité Técnico AEN/CTN139, es el organismo de estándares en España, y espejo del CEN a nivel nacional.
- ✓ **ISO, International Standards Organization** e **IEEE, Institute of Electrical and Electronics Engineering**, órganos de responsabilidad superior de los que dependen los comités internacionales y nacionales anteriores.

Las normas y estándares más destacados para interoperabilidad de sistemas de información en medicina son:

- ✓ **HL7, Health Level 7**, fundada por fabricantes americanos de equipos médicos, y acreditada por American National Standards Institute (ANSI), es un estándar para intercambio de mensajes médicos. Desarrolla una sintaxis propia, en los siete niveles de la pila de protocolos, para representar la información en una estructura sencilla compuesta por segmentos, tipos de datos y campos etiquetados.

- ✓ **EN13606**, especifica la arquitectura de información requerida para las comunicaciones interoperables entre sistemas y servicios que proveen o necesitan datos de la Historia Clínica Electrónica.

- ✓ **ISO/IEEE 11073 (X73)**, es una familia de normas, promovidas por IEEE, y adoptadas como estándar de ISO, y que agrupa diversas normas CEN anteriores para cubrir los diferentes niveles del modelo OSI.

La incorporación de todos los estándares existentes en un mismo sistema es complicado y requiere de un gran esfuerzo de integración. Para ello, es imprescindible la coordinación entre instituciones, empresas y otras organizaciones tanto sanitarias como de investigación.

Ante este panorama surgen otros dos organismos: ***Integrating the Healthcare Enterprise*** **(IHE)** [14] que trata de buscar, junto con los fabricantes de MDs, la mejor solución para cada servicio específico; y ***Continua Health Alliance*** [15], formada por 22 compañías del sector de tecnologías sanitarias, que persigue la incorporación de tecnologías interoperables en los dispositivos así como promover el uso de estos sistemas en las aplicaciones tanto a nivel profesional como cotidiano, planteando como reto la obtención de un certificado de normalización en forma de logotipo a incluir en los productos comerciales compatibles con los correspondientes estándares.

OpenECG es otra de estas fundaciones, ligada estrechamente a la difusión del estándar de electrocardiografía, SCP-ECG.

2.2 MDs y Android

Gracias a la telemedicina se han realizado muchos proyectos que permiten que una persona que se encuentra en un lugar remoto o sin el personal médico adecuado, pueda ser atendida en las mejores condiciones posibles por un especialista a distancia. Pero lejos de quedarse sólo en esos proyectos, los ingenieros que trabajan en esta rama han creado aplicaciones que permiten monitorizar el ritmo cardíaco de los pacientes e informar a los

médicos a través de un teléfono móvil, o crear cinturones que avisen cuando el paciente sufre una caída, o incluso controlar los niveles de glucosa en sangre de un enfermo de diabetes.

Dando un paso adelante en la tecnología, un equipo de ingenieros del Masschusetts Institute of Technology (MIT) decidieron aprovechar el entorno de programación que ofrece Android para crear una aplicación que permita mejorar el diagnóstico remoto y los servicios terapéuticos en las áreas más alejadas de la población. Y así nació el programa "Moca", un proyecto *open source* que ahora mismo se está probando en Filipinas para el telediagnóstico [16].

Moca se basa en una aplicación para Android que se comunica con un servidor Linux del que obtiene información médica concerniente a los pacientes, previa introducción de una serie de datos a en un sencillo formulario a través de un HTC G1 (*Google Dev Phone*). Para poder llevar a cabo la comunicación entre el servidor Linux y Android, se ha desarrollado un *plugin* específico que permite la transmisión multimodal de la información aprovechando al máximo los recursos de red disponibles en cada momento. Así, por ejemplo para enviar pequeñas anotaciones de texto el teléfono utiliza el protocolo de mensajes cortos, mientras que para comunicar imágenes o informes de mayor tamaño el teléfono utiliza una conexión GPRS. Además, para evitar problemas de retransmisión con las pérdidas de paquetes que pueden retardar mucho las transmisiones, Moca incorpora un protocolo que divide la imagen en pequeños paquetes permitiendo retomar una transmisión en el momento en que se cortara.

En numerosas zonas del planeta pueden pasar semanas hasta que un especialista pueda llevar a cabo un diagnóstico del paciente. Además, muchos pacientes deben realizar viajes extremadamente largos, de hasta medio día de duración, para llegar el centro médico más cercano, por lo cual no vuelven para hacer el seguimiento pertinente, o vuelven únicamente cuando sus condiciones son demasiado malas para tener solución.

Utilizando esta aplicación sobre un HTC G1 o cualquier otro dispositivo basado en Android, el personal médico que se desplaza a las zonas mencionadas, puede realizar un diagnóstico *insitu* en el mismo momento. Permite tomar una foto, adjuntar un archivo de voz, o incluso una radiografía si está disponible. Los datos son enviados a través de WiFi o GPRS a un servidor Linux (*Moca Dispatch Server*), donde se almacenan en una base de datos (ver Figura 2.1). Los especialistas analizan el material recibido y realizan un diagnóstico, que envían inmediatamente de vuelta al personal clínico desplazado a la zona.

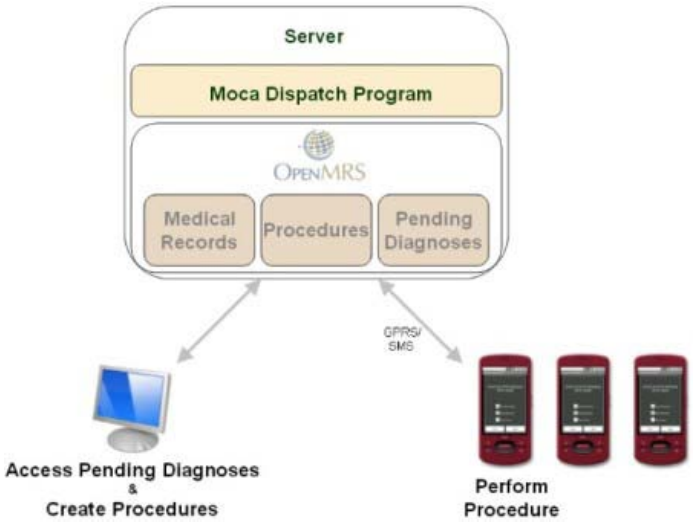


Figura 2.1 – Arquitectura del proyecto MOCA

El equipo de Moca eligió Android por tratarse de un sistema *open source* y multitarea con una cámara de calidad y que ofrece un avanzado entorno de desarrollo. Sin embargo, el elevado precio del HTC G1 suponía un problema en las áreas donde se iba a lanzar (Filipinas), ya que existían otros dispositivos *open source* pero las prestaciones que ofrecían no eran las requeridas. Finalmente, gracias a la colaboración y las ayudas recibidas por parte de la ONG One Laptop per Child (OLPC) el proyecto se ha podido llevar a la realidad [17].

Sin duda una pequeña muestra de las posibilidades que se abren para las telecomunicaciones gracias al uso de entornos abiertos para las aplicaciones.

2.3 Dispositivos médicos con X73

El intercambio de información y la interoperabilidad se puede entender como una serie de enormes beneficios para los sistemas sanitarios y los programas de telemedicina; los pacientes tendrán una mejor información sobre su estado de salud y los médicos tendrán la capacidad de controlar los signos vitales con mucha más facilidad.

En Marzo de 2009, Continua Health Alliance publicó el primer producto Certificado Continua™ en el mercado mundial. Nonin Medical [18] dio a conocer a través de internet su Pulsioxímetro Portátil Continua Certificado con puerto USB, **2500 PalmSAT®** (ver Figura 2.2).

Figura 2.2 – Modelo 2500 PalmSAT®

Nonin Medical, Inc. es una empresa privada con sede en Minneapolis que está especializada en el diseño y fabricación de soluciones de monitorización fisiológica no invasiva. Es de las primeras empresas de la industria en capacidad de procesamiento de señales y diseño de sensores, además integra características no disponibles en los productos de otras empresas. Ha sido un factor clave en la arquitectura de normas de interoperabilidad, incluida la norma ISO/IEEE 11073, USB Personal Health Device Class (PHDC) y HDP.

Hoy en día existen varios dispositivos médicos certificados por Continua, lo que pone de manifiesto el interés de la industria por este sector y, especialmente, por el desarrollo de equipos que cumplan con la norma X73. Excepto para la especialización del termómetro, existen dispositivos comerciales para el resto de implementaciones de este proyecto (tensiómetro, pulsioxímetro y báscula), todos ellos además con tecnología Bluetooth.

A continuación se muestran algunos ejemplos de los equipos disponibles para las especializaciones desarrolladas, todos ellos certificados por Continua, y que podrían ponerse en funcionamiento con la plataforma desarrollada en este PFC. Los dos primeros han sido fabricados por OMRON [19], mientras que el último pertenece a Nonin.

- **OMRON Bluetooth® Home Blood Pressure Monitor:** Se trata de un tensiómetro con tecnología Bluetooth que permite a los usuarios, además de tomar las medidas, transmitirlas fácilmente a otros sistemas electrónicos (ver Figura 2.3).

Figura 2.3 – OMRON Blood Pressure Monitor

Communication function: Bluetooth® Ver2.1+EDR Class2 (10m)
Profile : SPP/HDP (conforming to Continua design guidelines)
Data format standard : IEEE11073-10407
Pairing : Secure Simple Pairing/Passcode system
Data transmission : Measurement date/time, measurement values, model, and serial number

- **OMRON Weighing Scale:** Báscula certificada por Continua con tecnología inalámbrica Bluetooth que ofrece a los usuarios una visión amigable de su nivel de salud a través de varios indicadores (ver Figura 2.4).

Figura 2.4 – OMRON Weighing Scale

Communication function : Bluetooth® Ver2.1+EDR Class2 (10m)
Profile : SPP/HDP (conforming to Continua design guidelines)
Data format standard : IEEE11073-10415
Pairing : Secure Simple Pairing/Passcode system
Data transmission : Measurement date/time, measurement values, model, personal target values, serial number, and personal profile number

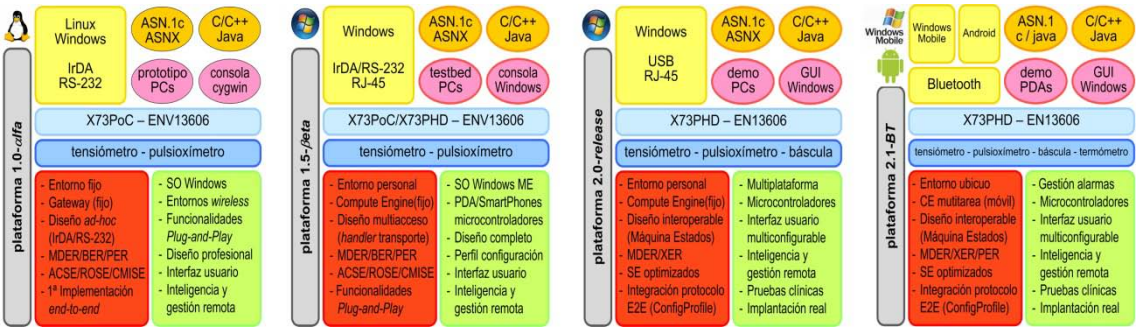
- **Nonin Onyx® II 9560 Wireless Fingertip Pulse Oximeter:** Pulsioxímetro certificado por Continua con tecnología Bluetooth que ofrece un intercambio de información simple y seguro, y que permite monitorizar a los pacientes en su vida diaria (ver Figura 2.5).

Figura 2.5 – Nonin Pulse Oximeter

Communication function : Bluetooth® Ver2.0+EDR Class1 (100m)
Profile : SPP/HDP (conforming to Continua design guidelines)
Data format standard : IEEE11073-10404
Pairing : Secure Simple Pairing/Passcode system
Data transmission : Measurement date/time, measurement values (SpO_2 + pulse rate), model, serial number

2.4 Evolución de la plataforma de telemonitorización

La evolución sufrida en los últimos años por la norma X73, así como el afán por llegar a un mayor número de personas a través de diferentes sistemas, ha hecho que la plataforma de telemonitorización original desarrollada en el grupo X73Spain [20], haya pasado por diversas migraciones, tal y como se expone a continuación (ver Figura 2.6).



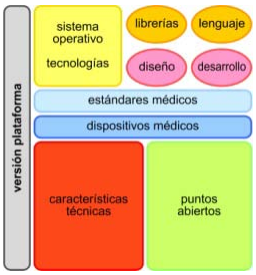


Figura 2.6 – Evolución de la plataforma de monitorización

2.4.1 Plataforma 1.0 – alfa

La plataforma 1.0-alfa es la **primera solución** *end-to-end* dividida en dos subsistemas: el subsistema de adquisición que permite la conexión (vía RS-232 e IrDA) entre los dispositivos médicos y un elemento central (*gateway*) conforme a X73PoC, y el subsistema de almacenamiento que soporta el intercambio de la información médica en un servidor de HCE conforme a ENV13606.

Se implementó una primera solución basada en leguajes C/C++ y Java para comunicar un pulsioxímetro DATEX-OHMEDA (no compatible con X73) con un *gateway* y un servidor de telemonitorización usando ordenadores personales operando sobre Linux (ver Figura 2.7).

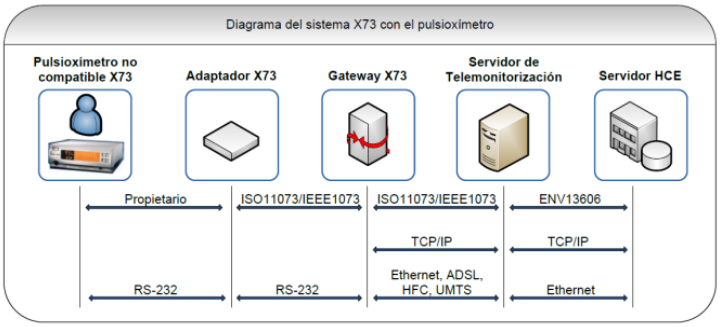


Figura 2.7 – Plataforma 1.0 – alfa

Las comunicaciones adaptador-*gateway* y *gateway*-servidor se llevaban a cabo mediante X73, mientras que la comunicación entre el dispositivo y el adaptador no cumplía el estándar. Los datos médicos eran almacenados según las estructuras DIM (*Domain Information Model*) definidas en la norma X73.

2.4.2 Plataforma 1.5 – beta

En esta plataforma se aborda el ámbito de la tele-asistencia sanitaria y la telemedicina; en concreto, la optimización de la norma X73 de interoperabilidad de dispositivos médicos.

Se mantienen los dos subsistemas de la versión anterior, pero evoluciona de X73-PoC a X73-PHD: independizando la capa de transporte (mediante un *handler* compatible con TCP/IP sobre USB y Bluetooth), incluyendo envío de datos episódico (el anterior periódico) mediante un sistema de *buffers*, y optimizando el *gateway* hacia el concepto de *Compute Engine* (CE) sobre una nueva máquina de estados finita.

Se migró del sistema operativo Linux a Windows, debido a su uso generalizado y a las mayores posibilidades y funcionalidades que ofrece en el desarrollo de sistemas *end-to-end* basados en la norma X73-PHD. Esta nueva versión 1.5-beta, implementa la parte de la comunicación entre un dispositivo médico (MD) y un PC actuando de *gateway* (CE).

Para el desarrollo de esta plataforma se dispuso de un tensiómetro OMRON modelo 7051T BT. Es un modelo de brazo donde la medida se realiza de forma fácil e intuitiva igual que en otros dispositivos convencionales. La diferencia radica en que este modelo está equipado con un interfaz *wireless* que transfiere automáticamente el valor medido a un archivo central, y requiere la instalación de un *software* en el PC para recibir los datos de las medidas.

2.4.3 Plataforma 2.0 – release

La segunda versión basada plenamente en X73-PHD, se encarga de solucionar los problemas de la anterior versión. Está orientada a entornos ubicuos y dispositivos llevables y a la aplicación de nuevas funcionalidades *plug&play* y de gestión remota. Implementa el protocolo de comunicación entre MD-CE, particularizado en X73-PHD.

Esta plataforma integra los dos subsistemas mediante un nuevo protocolo extremo a extremo, incluye todas las evoluciones tanto de X73-PHD como de EN13606, e incorpora un nuevo GUI. El código está optimizado respecto a la versión anterior (Plataforma 1.5 – beta) e incorpora funcionalidades para permitir su aplicación en soluciones de e-Salud. Consiste, a grandes rasgos, en la implementación de la norma sobre un sistema compuesto por varios MD y un CE.

2.4.4 Plataforma 2.1 – BT

Esta plataforma mantiene la estructura de la pila y los subsistemas de las anteriores versiones, y se basa en modificar el *software* heredado de la Plataforma 2.0 – release para adaptarlo a dispositivos móviles con tecnología Bluetooth, concretamente a *smartphones* con sistema operativo Windows Mobile 6 (ver Figura 2.8), optimizando las funcionalidades ubicuas

(*plug-and-play* y *hot-swap*). De este modo, fue necesario sustituir las librerías de las que se disponía en la anterior plataforma para poder realizar un *software* compatible.

Figura 2.8 – Smartphone CE y MD

3. Análisis y diseño

En este capítulo se estudian los requisitos que debe cumplir el proyecto, que coinciden, principalmente, con las necesidades de los pacientes que en diferentes situaciones tendrán que tomarse una serie de medidas clínicas; se expondrá con detalle el diseño realizado y la arquitectura implementada.

3.1 Análisis de requisitos

En este apartado se analizan las necesidades de los pacientes, usuarios potenciales del *software*, para determinar qué debe hacer el sistema a desarrollar.

Existen tres tipos de escenarios en los que se puede situar al paciente que precisa de uno de los dispositivos médicos implementados: Domiciliario, Hospitalario y Fitness.

- ✓ **Domiciliario**: Son pacientes individuales, con un número reducido de dispositivos médicos, todos ellos dentro de una red PAN (*Personal Area Network*). La configuración del sistema se compondrá de MDs con interfaz Bluetooth, un CE también Bluetooth y una conexión WiFi o 3G.

- ✓ **Hospitalario**: En este caso el que hace uso del MD es personal sanitario experto, por lo tanto, asociado a cada usuario, habrá un número variable de pacientes. El sistema se configurará en este caso con MDs y CE wireless y una conexión WiFi.

- ✓ **Fitness**: Este es un caso particular de los anteriores pero con una interfaz de usuario más atractiva y comercial.

Tras haber detectado la necesidad de basarse en un estándar para dotar de interoperabilidad al proyecto, se establecerá como guía la norma X73 y se seguirán sus pautas para el diseño del *software*. Mediante una máquina de estados finitos (FSM) la norma describe cómo sincronizan su funcionamiento un sistema agente-manager. El diseño de la FSM (ver

Figura 3.1), es la clave de cualquier solución basada en X73 dado que define el mecanismo principal de comportamiento.

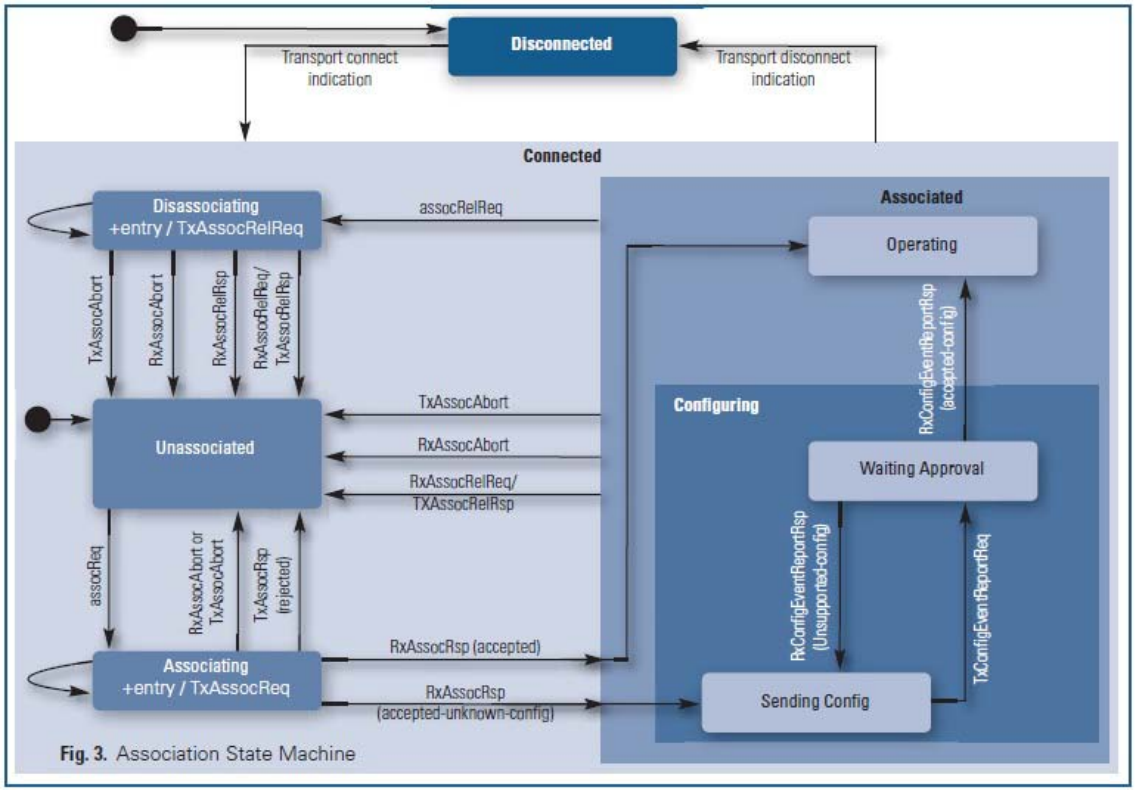


Figura 3.1 – Máquina de estados FSM genérica

Para el diseño se han de tener en cuenta los estados definidos en la norma X73: *DISCONNECTED, CONNECTED, UNASSOCIATED, ASSOCIATED, CONFIGURING* y *OPERATING*. El proceso de funcionamiento, (ver Figura 3.2), sería:

- Ambos equipos son encendidos, desencadenando el proceso de inicialización local.
- A partir de esta inicialización, se establece una conexión a través de la capa de transporte; si tiene éxito, ambos pasan al estado *CONNECTED*, pero *UNASSOCIATED*. Para asociarse, el agente envía una petición de asociación al manager (*Association Request*).
- Si el manager conoce la configuración del agente, bien porque es estándar o bien porque la tiene almacenada de operaciones anteriores, ambos pasan a *ASSOCIATED* y podrán operar. Si no, el manager solicitará previamente la configuración del agente (estado *CONFIGURING*) y la podrá almacenar para otras ocasiones (esto facilita las funcionalidades *plug&play*).

- Una vez alcanzado el estado *OPERATING*, se inicia el envío de muestras. La forma en la que el agente realiza el envío, varía en función del dispositivo y de la medida en cuestión, puede realizar un solo envío, o envíos sucesivos durante un periodo de tiempo limitado o ilimitado.

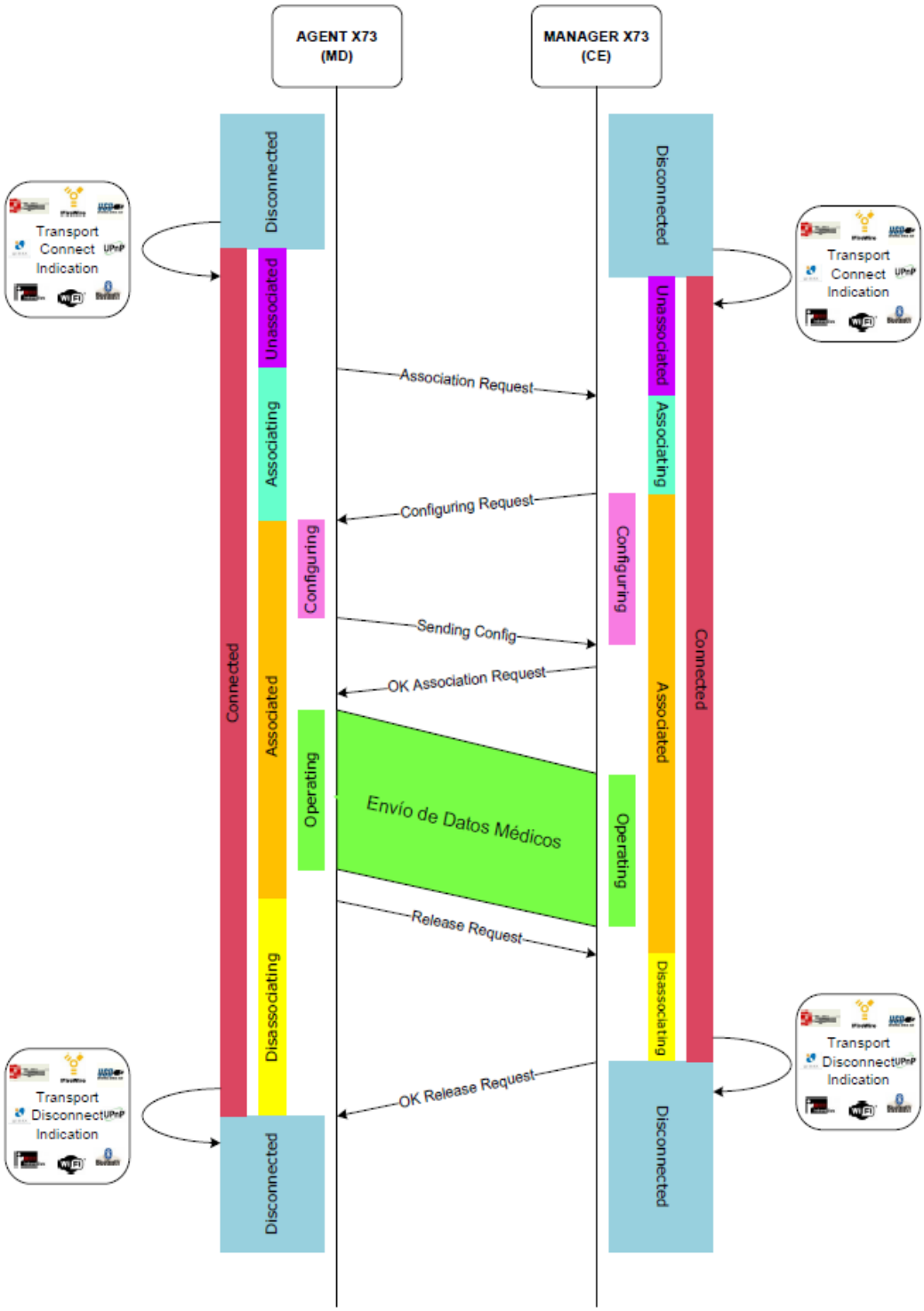


Figura 3.2 – Intercambio de tramas entre el MD y el CE

- En cualquier momento se pueden desasociar tanto agente como manager: en situaciones de error, por finalización del envío de medidas, o por otras circunstancias. Para ello se dispone de una solicitud de desasociación (*Release Request*) seguida de la confirmación del otro extremo o una desasociación directa.

Una vez analizados los requisitos fundamentales para el desarrollo del proyecto, tanto teóricos/técnicos (norma X73) cómo prácticos (casos de uso), se debe diseñar un esquema de la solución adoptada. Inicialmente, el planteamiento fue el de conectar una serie de dispositivos médicos (agentes) con un dispositivo móvil (manager) a través de tecnología Bluetooth y siguiendo la norma X73 (ver Figura 3.3).

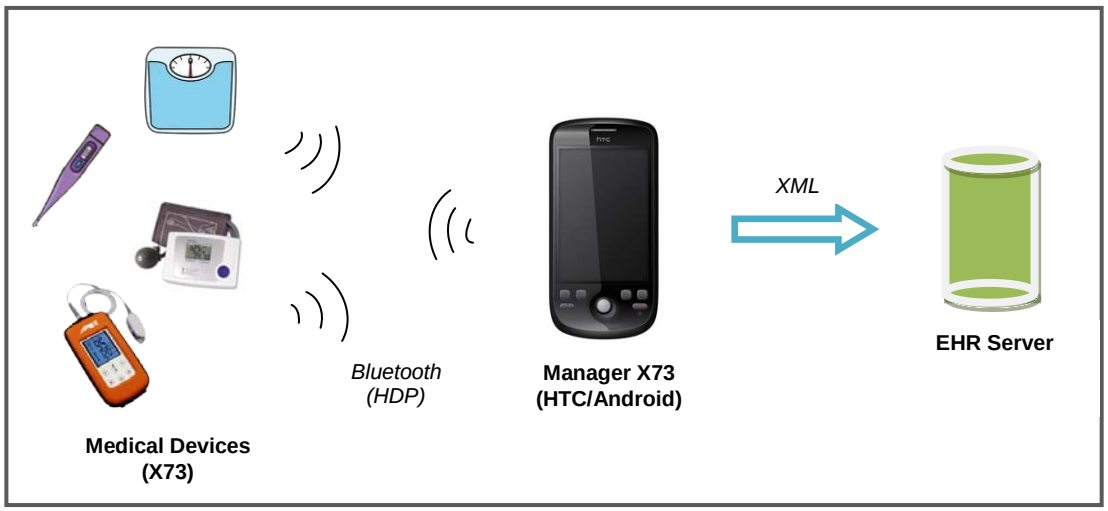


Figura 3.3 – Esquema de la solución adoptada

Sin embargo, esta solución debe ser adaptada a los recursos disponibles. Por lo tanto, al no disponer de dispositivos médicos reales con interfaz Bluetooth y que cumplan el estándar, se optó por simularlos con un terminal móvil, de la misma forma que el manager se implementa sobre un terminal del mismo tipo.

Así, para el desarrollo del proyecto se han utilizado dos dispositivos HTC G2 con sistema operativo Android, con los que se simulan tanto los MDs como el CE, a través de programación en lenguaje Java. Se tendrá, por un lado, el **X73PHDAgent**, que implementa varios MDs (Termómetro, Tensiómetro, Pulsioxímetro y Báscula), y por otro, el **X73PHDManager**. Los motivos por los cuales se tomó la decisión de utilizar estos terminales puede verse, junto con una breve introducción a Android, en el Anexo B.

3.2 Especificación

La norma X73, a través de sus diversas especializaciones IEEE 11073 – 104xx [7], determina el modelo DIM de cada uno de los dispositivos soportados. En este proyecto se han implementado las especializaciones correspondientes al termómetro (10408), tensiómetro (10407), pulsioxímetro (10404) y báscula (10415). Cada una de estas especializaciones, establece tanto el modelo de comunicación, como la configuración y el formato de los datos médicos utilizados en cada caso (temperatura, tensión arterial, pulso, saturación de oxígeno en sangre y peso).

A continuación se explica más detalladamente la especialización correspondiente al termómetro, si bien para el resto de dispositivos implementados la estructura es similar, con ciertas particularidades. Se diferencian básicamente en el tipo de datos médicos con los que se trabaja, por lo que varían algunos atributos: en el termómetro y en la báscula, por ejemplo, se envía solamente un objeto medida (temperatura y peso respectivamente), mientras que para el pulsioxímetro se envían dos (saturación de oxígeno en sangre y pulso), al igual que ocurre con el tensiómetro (un objeto para la tensión sistólica, diastólica y MAP, y otro para el pulso).

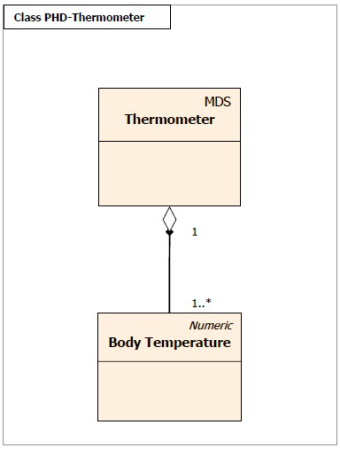


Figura 3.4 – Termómetro, Domain Information Model

Como puede verse en la Figura 3.4, aparece por un lado el objeto MDS (*Medical Device System*) y por otro, dependiendo de éste último, el objeto correspondiente a la medida en cuestión, en este caso, la temperatura (*Body Temperature*). Para cada uno de ellos, la norma determina los atributos que los componen, con su correspondiente valor y si éstos son obligatorios o no, diferenciando para el caso de configuración estándar y para el de configuración extendida (ver Tabla 3.1, donde aparecen resaltados los atributos de mayor importancia).

Tabla 3.1 – Atributos del objeto MDS Thermometer

Attribute name	Extended Configuration			Standard Configuration		
	Value	Ctxt.	Qual.	Value	Ctxt.	Qual.
Handle	0	Static	M	0	Static	M
System-Type	MDC_DEV_SPEC_PROFILE_TEMP	Static	M	MDC_DEV_SPEC_PROFILE_TEMP	Static	M
System-Model	"Manufacturer","Model"	Static	M	"Manufacturer","Model"	Static	M
System-Id	EUI-64	Static	M	EUI-64	Static	M
Dev-Configuration-Id	0x800–0x899	Static	M	0x800	Static	M
Attribute-Value-Map	See 0	Static.	C	See 0	Static	C
Production-Specification	See 0	Static	O	See 0	Static	O
Mds-Time-Info	See 0	Static	C	See 0	Static	C
Date-and-Time	See 0	Dyn.	C	See 0	Dyn.	M
Relative-Time	See 0	Dyn.	C	See 0	Dyn.	X
HiRes-Relative-Time	See 0	Dyn.	C	See 0	Dyn.	X
Power-Status	*onBattery* or *onMains*	Dyn.	R	*onBattery* or *onMains*	Dyn.	R
Battery-Level	See 0	Dyn.	R	See 0	Dyn.	R
Remaining-Battery-Time	See 0	Dyn.	R	See 0	Dyn.	R

- Mandatory (M): obligatorio
- Conditional (C): será requerido o no según alguna condición de estado
- Recommended (R): recomendado
- Optional (O): opcional
- Not Recommended (NR): no recomendado
- Not Allowed (X): no permitido

La diferencia entre la configuración estándar y la extendida es que mientras para la configuración estándar no es necesario un proceso de configuración (estado *Configuring* de la máquina de estados), éste sí que es requerido con la configuración extendida. Además, en el modo extendido, la norma permite incluir otras medidas en el mismo objeto donde antes sólo se incluía la temperatura y la fecha y hora, pudiendo añadir también la altura, el peso y el índice de masa corporal, si bien estos parámetros son opcionales y no se utilizan en la configuración básica.

De la misma forma, se tiene la tabla de atributos del objeto *Body Temperature*, el cual es de tipo *Numeric* y está identificado por el código de nomenclatura `MDC_MOC_VMO_METRIC_NU` (ver Tabla 3.2).

Tabla 3.2 – Atributos del objeto métrico Body Temperature

Attribute name	Extended Configuration			Standard Configuration (Dev-Configuration-Id = 0x800)		
	Value	Ctxt.	Qual.	Value	Ctxt.	Qual.
Handle	See 0	Static	Handle	1	Static	Handle
Type	MDC_PART_SCADA | See IEEE Std 11073-10101	Static	M	MDC_PART_SCADA | MDC_TEMP_BODY	Static	M
Metric-Spec-Small	0xD040	Static	M	0xD040	Static	M
Measurement-Status	See 0	Dyn.	R	See 0	Dyn.	O
Metric-Id	See 0	Static	X	See 0	Static	X
Unit-Code	MDC_DIM_DEGC or MDC_DIM_FAHR	Static	M	MDC_DIM_DEGC	Static	M
Attribute-Value-Map	See 0	Static	C	MDC_ATTR_NU_VAL_OBS_SIMP | MDC_ATTR_TIME_STAMP_ABS	Static	M
Source-Handle-Reference	See 0	Static	X	See 0	Static	X
Label-String	See 0	Static	O	See 0	Static	O
Unit-LabelString	See 0	Static	O	See 0	Static	O
Absolute-Time-Stamp	See 0	Dyn.	C	See 0	Dyn.	M
Relative-Time-Stamp	See 0	Dyn.	C	See 0	Dyn.	C
HiRes-Time-Stamp	See 0	Dyn.	C	See 0	Dyn.	X
Measure-Active-Period	See 0	Dyn.	X	See 0	Dyn.	X
Simple-Nu-Observed-Value	See ISO/IEEE P11073-20601	Dyn.	C	See ISO/IEEE P11073-20601	Dyn.	M
Basic-Nu-	See 0	Dyn.	C	See 0	Dyn.	C

3.3 Diseño y arquitectura

En este apartado se explica el diseño realizado para el desarrollo del *software*, así como la arquitectura establecida. En primer lugar, la implementación se ha llevado a cabo según principios fundamentales de modularidad, escalabilidad y abstracción de capas, es decir, separando los diferentes bloques que componen el programa, permitiendo de esta manera su exportación y utilización en otros entornos, y facilitando una posible ampliación futura.

De esta forma, se tiene por un lado la parte correspondiente a la norma X73, y por otro la correspondiente al código del propio programa, dividida esta última a su vez en interfaz gráfica, utilidades (conversión de tipos, definiciones…), eventos, y clases propias del sistema operativo Android. Además, particularizando al caso del X73PHDAgent, cada uno de los MDs soportados se ha implementado por separado, lo que permite añadir nuevos dispositivos de una forma rápida y sencilla, sin necesidad de modificar el resto del código (ver Figura 3.5).

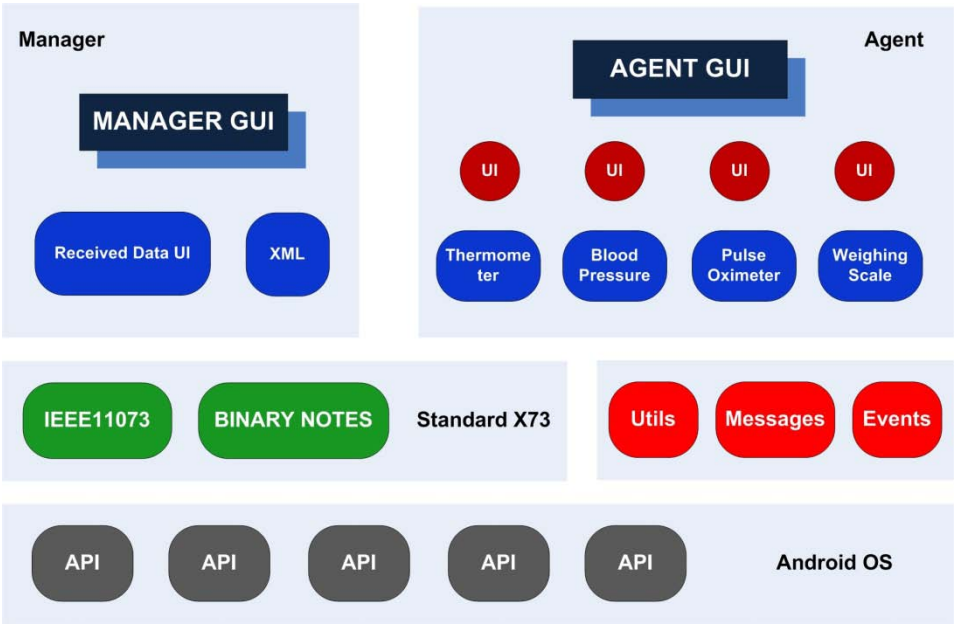


Figura 3.5 – Diagrama del software implementado (Agente y Manager)

A continuación se explican con mayor detalle cada uno de los bloques que componen la implementación, vistos en el diagrama anterior.

3.3.1 Binary Notes

El paquete Binary Notes [21] ofrece un *framework* de ASN.1 (*Abstract Syntax Notation One*) [22] para entornos Java y .NET, que contiene:

- *Encoding/Decoding Library*: Librería para la codificación/decodificación. Implementa las codificaciones BER (*Basic Encoding Rules*), DER (*Distinguished Encoding Rules*), y PER (*Packed Encoding Rules*).

- *BNCompiler:* Compilador ASN.1 que genera el código Java / C# para un archivo ASN.1 de entrada determinado.

- BinaryNotesMQ: Proporciona cadenas de mensajes basadas en la codificación ASN.1 que permiten la interoperabilidad de entornos Java y .NET.

ASN.1 es una notación formal utilizada para describir la transmisión de datos que se lleva a cabo a través de los diferentes protocolos de comunicación existentes. Además establece una implementación del lenguaje y una representación física de estos datos, sea cual sea la aplicación y su complejidad.

3.3.2 IEEE 11073

Este bloque implementa la norma X73 (máquina de estados, definiciones, nomenclaturas…). Está dividido a su vez en tres partes:

- **Parte 10101:** Aquí se definen todas las nomenclaturas utilizadas por la norma, es decir, todos los códigos que representan los diferentes atributos, tipos de medidas, de dispositivos, etc.

- **Parte 104xx:** En esta parte se implementan las especializaciones desarrolladas, *Thermometer* (10404), *Blood Pressure Monitor* (10407), *Pulse Oximeter* (10408) y *Weighing Scale* (10415), tanto en su configuración estándar como en la extendida.

- **Parte 20601:** En este bloque se establece el funcionamiento de la máquina de estados que rige el estándar, las definiciones ASN1 de todos los tipos de datos utilizados, y se define el PHD.

Lógicamente, este paquete no será igual para el manager y para los agentes, si bien su estructura es idéntica y en ambos casos se utiliza de la misma forma. Por ejemplo, en el caso del manager, incluye también el código correspondiente al *socket* Bluetooth que escucha y recibe las conexiones entrantes procedentes del agente.

3.3.3 Utils, Messages y Events

En estos paquetes se incluyen funciones útiles que sirven de gran ayuda a la hora de desarrollar el código principal:

- **Utils:** Aquí aparecen utilidades para trabajar con los datos definidos por ASN1, con los objetos DIM, y con las colas de datos usadas en la transmisión.
- **Messages:** Como su propio nombre indica, sirve para trabajar con los mensajes que se envían el agente y el manager.
- **Events:** Bloque muy importante que permite el lanzamiento de eventos “hacia arriba” (hacia la interfaz gráfica) para actualizar los datos oportunos en la pantalla del terminal, por ejemplo, cuando llega una nueva medida o cuando se pasa de un estado a otro.

3.3.4 Manager

Contiene la parte de más alto nivel del **X73PHDManager**, es decir, la interfaz gráfica principal y la pantalla donde se muestran los datos enviados por el agente (ver Figura 3.6).

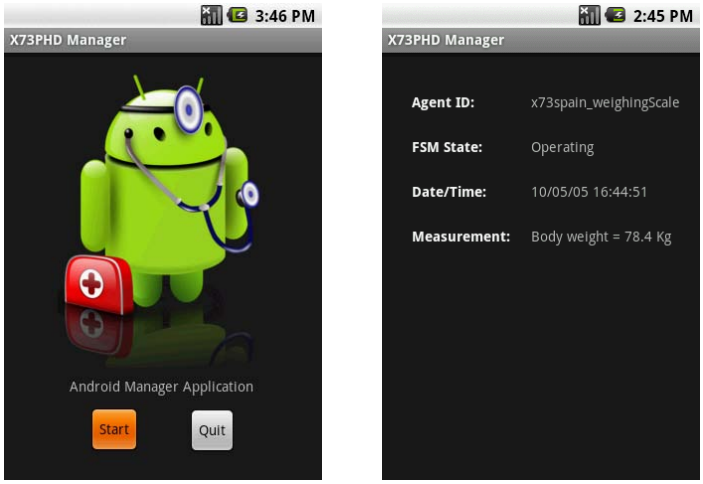


Figura 3.6 – a)Manager GUI y b)Received Data UI

Además, incluye la clase correspondiente al trabajo con los ficheros XML donde se almacenan, opcionalmente, los atributos correspondientes a las medidas recibidas. El formato de estos ficheros XML (ver Figura 3.7), se ha definido según lo establecido conjuntamente con otros miembros del grupo X73Spain, ya que, además de almacenarse de forma local, se pueden enviar a una base de datos de HCE, función incluida también en este bloque.

```
<?xml version="1.0" encoding="utf-8" ?>
- <report>
    <protocol>X73</protocol>
    <!-- INFORMACION SOBRE EL MDS  -->
  - <deviceInfo>
    - <attribute>
        <id>...</id>
        <value>...</value>
      </attribute>
    </deviceInfo>
    <!-- AQUI COMIENZA LA INFORMACION DE LA MEDIDA  -->
  - <measurement>
    - <attribute>
        <id>MDC_ATTR_ID_TYPE</id>
        <value>MDC_TEMP_BODY</value>
      </attribute>
    - <attribute>
        <id>MDC_ATTR_NU_VAL_OBS_SIMP</id>
        <value>37.5 ºC</value>
      </attribute>
    - <attribute>
        <id>MDC_ATTR_UNIT_CODE</id>
        <value>MDC_DIM_DEGC</value>
      </attribute>
    - <attribute>
        <id>MDC_ATTR_TIME_STAMP_ABS</id>
        <value>21</value>
        <value>10</value>
        <value>04</value>
        <value>30</value>
        <value>12</value>
        <value>42</value>
        <value>31</value>
        <value>00</value>
      </attribute>
    </measurement>
  </report>
```

Figura 3.7 – Formato ficheros XML

3.3.5 Agent

Al igual que en el caso del Manager (punto 3.3.4), en este apartado se encuentra la parte gráfica del **X73PHDAgent**, tanto de la pantalla principal, como de las especializaciones implementadas, donde se introducen las medidas que serán enviadas al manager. Como se ha explicado anteriormente, al no disponer de los dispositivos médicos reales, estas medidas no se toman físicamente, sino que se simulan con los terminales móviles, introduciéndolas manualmente en la pantalla de estos (ver Figura 3.8).

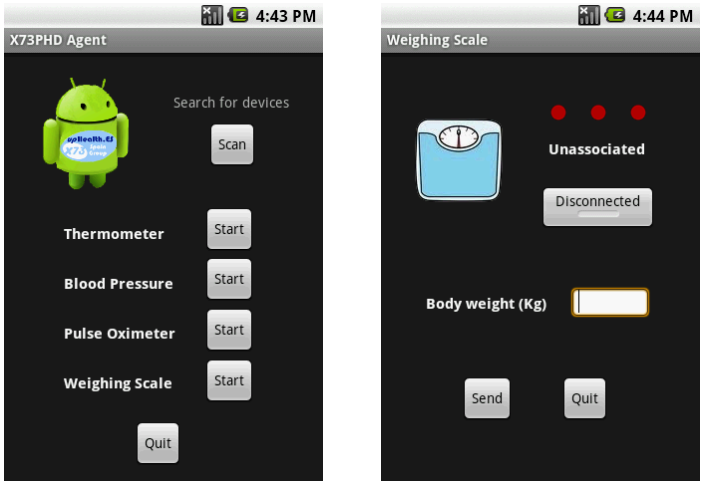


Figura 3.8 – a)Agent GUI y b)Weighing Scale UI

En la Figura 3.8 se muestra, además de la pantalla de inicio del **X73PHDAgent**, la correspondiente a la báscula, donde se ejecuta la conexión y se introduce la medida. Para el resto de especializaciones, la interfaz es equivalente, cambiando la(s) medida(s) a enviar y la imagen correspondiente.

Además de la parte gráfica, se incluye también el código referente a cada uno de los dispositivos: generación de datos y mensajes a enviar, chequeo de las respuestas provenientes del manager, máquina de estados, etc.

Por otro lado, cuenta con el código correspondiente a la conexión Bluetooth: descubrir dispositivos dentro de su alcance, parearse con ellos, y creación de los sockets de comunicación necesarios para el envío y recepción de datos entre agente y manager.

4. Desarrollo e implementación

Una vez conocido el diseño que se ha llevado a cabo en este proyecto, se pasará a mostrar el desarrollo a más bajo nivel, detallando las partes fundamentales del código y presentando aspectos importantes de la programación realizada; se explicará también el funcionamiento de la plataforma a nivel de usuario.

4.1 Estructura del código

Como ya se expuso en el capítulo 3.1, el código ha sido escrito en lenguaje Java, más concretamente en J2ME (*Java Mobile Edition*), a través del IDE (*Integrated Development Enviroment*) Eclipse, ya que éste cuenta con un *plugin* específico para el trabajo con Android, el cual proporciona herramientas de simulación a través de dispositivos virtuales (emulador), ayudas a la hora de diseñar las interfaces, consola para mostrar salidas por pantalla y controlar errores durante la depuración (modo *debug*), etc. Además del *plugin* comentado, se debe descargar el SDK de desarrollo de Android que proporciona el propio Google de forma gratuita (más información en el Anexo B).

Cada uno de los proyectos realizados con Eclipse, agente y manager, se distribuye en varias carpetas de la siguiente manera (ver Figura 4.1):

- **bin:** aquí se guardan los archivos binarios generados durante la compilación.
- **gen:** donde se crean ficheros Java generados automáticamente.
- **res**: contiene recursos como imágenes o diseños de las interfaces (*layouts*).
- **src**: donde se encuentra el código fuente en cuestión, que a su vez está compuesta por las carpetas: **es** (código principal), **ieee_11073** (norma X73), y **org** (Binary Notes).
- Finalmente, existe además un archivo llamado **AndroidManifest.xml** que establece la información de la aplicación a tener en cuenta por el compilador (permisos, versión del SDK…).

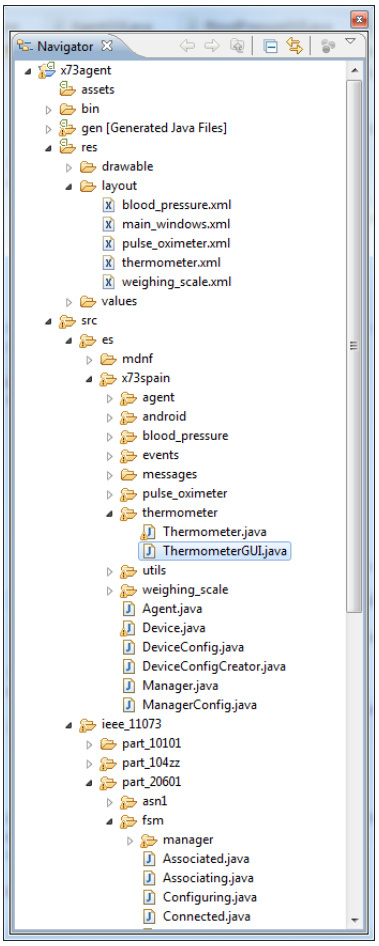


Figura 4.1 – Estructura del código en Eclipse

4.2 Programación

Se explica ahora de forma más detallada el código desarrollado, empezando por el X73PHDAgent, y particularizado al caso del termómetro, si bien la explicación es extensible al resto de especializaciones.

La primera parte a destacar en el código del agente, es la correspondiente a la conexión Bluetooth, que se encuentra en las clases “BluetoothActivity.java” y “DeviceListActivity.java”, encargadas de descubrir los dispositivos dentro de su alcance y conectarse con el manager. Una vez realizada la conexión, se podrá crear un *socket* Bluetooth entre ambos dispositivos para llevar a cabo la comunicación. Este bloque se ejecuta pulsando el botón **Scan** que aparece en la pantalla principal del X73PHDAgent (ver Figura 3.8a).

Al tratarse de un diseño modular, cada especialización está implementada por separado, lo que facilita futuras ampliaciones o modificaciones del programa. Así pues, para cada

dispositivo, se tiene una clase correspondiente al interfaz gráfico (“ThermometerGUI.java”, por ejemplo) y otra clase donde se gestionan los mensajes de la FSM enviados y recibidos (en el caso del termómetro sería “Thermometer.java”).

De esta forma, cuando se pulsa el interruptor **Connected/Disconnected** (ver Figura 3.8b), en la clase “ThermometerGUI” se crea una instancia de la clase “Thermometer”, y se inicia automáticamente el proceso de asociación entre agente y manager. Para ello, el primer paso es enviar el *AssociationRequest*, para después evaluar el *AssociationResponse* enviado por el manager y pasar o no al estado *Associated* (ver Figura 4.2).

Es importante destacar el cometido de la función “InternalEventReporter”, encargada de enviar eventos para la actualización del interfaz gráfico, que serán recibidos por un “InternalEventManager” de la clase “ThermometerGUI”.

```
s = device.createRfcommSocketToServiceRecord(MY_UUID);
s.connect();
//s = new Socket(managerIP, managerPort);

/** STATE: CONNECTED - ASSOCIATING **/

//Create Association Request
createAssociationRequest(thermType);
InternalEventReporter.agentChangeStatus(thermID, "Associating");

//Send Association Request  -->  ASSOCIATING
sendApdu(apduAsReq, s);

//Get Association Request Response
apduAsReqRsp = getApdu(s);

//Check Response
evalueAssociationResponse(apduAsReqRsp, s);

if(associated){
    /** STATE: CONNECTED - ASSOCIATED**/
    //Send event to update agent state
    InternalEventReporter.agentChangeStatus(thermID, "Operating");
}else{
    //Error
    InternalEventReporter.agentConnectionEvent(thermID, connError);
}
```

Figura 4.2 – Código del proceso de asociación

Una vez que ambos dispositivos están asociados y en modo *Operating* se procede al envío de los datos médicos. Pulsando sobre el botón **Send** (ver Figura 3.8b) se toma la medida introducida en el interfaz gráfico y se inicia el envío de estos datos (ver Figura 4.3).

```
//Create and Send Data
createData(s);
sendApdu(apduDataReq,s);

//Get Data Response
apduDataRsp = getApdu(s);

//Check Data Response
evaluePresentationResponse(apduDataRsp, s);
if(measureSentOK){
    InternalEventReporter.agentConnectionEvent(thermID, "Measure Sent OK");
}else{
    InternalEventReporter.agentConnectionEvent(thermID, connError);
}
```

Figura 4.3 – Código del envío de datos médicos

De la misma forma, cuando se pulse el botón **Quit** para finalizar la conexión, tendrá lugar el proceso de desasociación, volviendo de nuevo al estado *Disconnected* (ver Figura 4.4).

```
//Create Association Release Request
createAssociationReleaseRequest(s);

//Send Association Release Request
sendApdu(apduRelReq, s);

//Check Association Release Response
apduRelRsp = getApdu(s);
evalueAssociationReleaseResponse(apduRelRsp, s);

if(normalRelease){
    try{
        s.close();
    }
    catch(Exception e){
        System.out.println("finishThermometer exception: " + e.toString());
    }
}else{
    InternalEventReporter.agentConnectionEvent(thermID, connError);
}
```

Figura 4.4 – Código del proceso de desasociación

Hasta aquí se han mostrado las partes principales del código del agente para el caso del termómetro. Para ver con más detalle las funciones que aparecen y el resto del código, ver el CD adjunto. Se pasa ahora a explicar el código desarrollado para el manager.

En este caso, en lugar de los bloques de cada especialización que había para el agente, se tendrá solamente un bloque correspondiente al código principal del manager, donde se gestionan los interfaces y se recibirán las medidas enviadas por los dispositivos médicos. Habrá entonces un archivo “ManagerGUI.java” correspondiente a la pantalla principal, y otro “Manager.java” donde llegarán las medidas y se mostrarán en la interfaz.

Al igual que antes, los eventos que puedan llegar desde el agente se recibirán en un "InternalEventManager", lo que permitirá cambiar el estado, representar los datos médicos, etc. En la Figura 4.5 puede verse una parte de esta función.

```
private InternalEventManager ieManager = new InternalEventManager(){

    @Override
    public void agentChangeStatus(String systemId, String state) {
        agentState = state;
        handler.post(doUpdateState);
        System.out.println("ID: " + systemId + " state: " + state);
    }

    @Override
    public void agentConnected(Agent agent) {
        deviceName = agent.getSystem_id() + " (" + config_id + ")";
        handler.post(doUpdateGUI);
    }

    @Override
    public void agentDisconnected(String systemId) {
        agentState = "Disconnected";
        //deviceName = "No Agent";
        measure1 = "";
        measure2 = "";
        measure3 = "";
        measure4 = "";
        date = "";
        handler.post(doUpdateState);
        System.out.println("Agent " + systemId + " disconnected");
        //finaliceManager();
    }

    @Override
    public void receivedMeasure(String systemId, List measures) {

        System.out.println("Measures received from " + systemId);

        Iterator i = measures.iterator();
        if (measures.isEmpty()) return;

        while (i.hasNext()) {

            if ((400 <= config_id) && (config_id <= 499)){      // Pulse Oximeter

                measure1aux = getMeasure(i.next().toString(), " %");
```

Figura 4.5 – Código InternalEventManager en "Manager.java"

Otra parte destacada de esta clase "Manager" es la función "saveDataToXml", encargada, como su propio nombre indica, de almacenar los datos médicos recibidos en un fichero XML. En esta función se creará el archivo con el nombre del agente y la fecha, y se creará una instancia de la clase "XML.java", que se encuentra en el bloque Android, y que se ocupa de escribir los datos con el formato XML establecido en el fichero previamente creado. De la misma forma existe una función "sendData" que se ocupa de enviar el fichero XML generado a un servidor de HCE.

Además de la parte de más alto nivel explicada, es importante también comentar la parte correspondiente a las especializaciones definidas por el estándar. Estas se encuentran en la carpeta "part_104zz", dentro del bloque "ieee_11073". Existe una clase llamada

"DeviceSpecializationFactory", donde se crean las instancias de cada dispositivo, estableciendo los atributos requeridos para cada MD. Por otro lado, hay una clase para cada una de las especializaciones: "DS_10408.java" (termómetro), "DS_10407.java" (tensiómetro), "DS_10404.java" (pulsioxímetro), y "DS_10415.java" (báscula). En ellas se crean los objetos correspondientes a los datos médicos, con los atributos oportunos en cada caso según establece la norma X73 (ver Figura 4.6).

```
public static DS_10408 getThermometer10408 (byte[] system_id, ConfigId devConfig_id){
    DS_10408 thermometer = null;

    Hashtable<Integer,Attribute> mandatoryAttributes = new Hashtable<Integer,Attribute>();
    try {
        //from Part 10408: Handle=0
        HANDLE handle = new HANDLE();
        handle.setValue(new INT_U16(new Integer(0)));
        mandatoryAttributes.put(Nomenclature.MDC_ATTR_ID_HANDLE,
                new Attribute(Nomenclature.MDC_ATTR_ID_HANDLE,
                        handle));

        //from Part 10408: {"Manufacturer","Model"}
        SystemModel systemModel = new SystemModel();
        systemModel.setManufacturer("Thermometer".getBytes("ASCII"));
        systemModel.setModel_number("10408".getBytes("ASCII"));
        mandatoryAttributes.put(Nomenclature.MDC_ATTR_ID_MODEL,
                new Attribute(Nomenclature.MDC_ATTR_ID_MODEL,
                        systemModel));

        mandatoryAttributes.put(Nomenclature.MDC_ATTR_SYS_ID,
                new Attribute(Nomenclature.MDC_ATTR_SYS_ID,
                        system_id));

        mandatoryAttributes.put(Nomenclature.MDC_ATTR_DEV_CONFIG_ID,
                new Attribute(Nomenclature.MDC_ATTR_DEV_CONFIG_ID,
                        devConfig_id));

        //from Part 10408: {MDC_DEV_SPEC_PROFILE_TEMP,1}
        TypeVerList syst_type_data_list = new TypeVerList();
        syst_type_data_list.initValue();
        TypeVer item = new TypeVer();
        OID_Type oid = new OID_Type();
        oid.setValue(new INT_U16(new Integer(Nomenclature.MDC_DEV_SPEC_PROFILE_TEMP)));
        item.setType(oid);
        item.setVersion(new Integer(1));
        syst_type_data_list.add(item);
        mandatoryAttributes.put(Nomenclature.MDC_ATTR_SYS_TYPE_SPEC_LIST,
                new Attribute(Nomenclature.MDC_ATTR_SYS_TYPE_SPEC_LIST,
                        syst_type_data_list));
        thermometer = new DS_10408 (mandatoryAttributes);

    } catch (InvalidAttributeException e) {/*Never thrown in standadard configuration*/
        e.printStackTrace();
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
    return thermometer;
```

Figura 4.6 – Código de la especialización del termómetro

Por último, destacar también el código correspondiente a la creación del *socket* Bluetooth donde el manager recibirá las posibles conexiones de los agentes, y que se encuentra en la clase "BluetoothManagerChannel" (ver Figura 4.7).

```
public class BthManagerChannel extends Thread {
   private boolean finish = false;
   //private ServerSocket ss;
   private BthManagedAgents agents = new BthManagedAgents();

   private final BluetoothServerSocket mmServerSocket;
   private BluetoothAdapter mAdapter;

   private static final UUID MY_UUID = UUID.fromString("ff8f9fbc-ba07-4ba5-bcba-9eac678d1704");
   private static final String NAME = "X73PHDAndroid";

   public BthManagerChannel() {
       // Use a temporary object that is later assigned to mmServerSocket,
       // because mmServerSocket is final
       BluetoothServerSocket tmp = null;
       try {
           mAdapter = BluetoothAdapter.getDefaultAdapter();
           // MY_UUID is the app's UUID string, also used by the client code
           tmp = mAdapter.listenUsingRfcommWithServiceRecord(NAME, MY_UUID);
       } catch (IOException e) {
           System.out.println("Error: " + e.getMessage());
       }
       mmServerSocket = tmp;
   }


   public void run() {
       BluetoothSocket socket = null;
       // Keep listening until exception occurs or a socket is returned
       while (true) {
           System.out.println("Waiting for clients...");
           try {
               socket = mmServerSocket.accept();
           } catch (IOException e) {
               System.out.println("Error: " + e.getMessage());
               break;
           }
           // If a connection was accepted
           if (socket != null) {
               // Do work to manage the connection
               BthChannel chnl = null;
               try {
                   chnl = new BthChannel (socket);
                   Agent a = new Agent();
                   a.addChannel(chnl);
                   agents.addAgent(a);
```

Figura 4.7 – Código del socket Bluetooth abierto por el manager

4.3 Funcionamiento del programa

A continuación se explica el funcionamiento del programa, haciendo referencia a las funciones y operaciones más importantes.

Primeramente, se lanza el ejecutable que aparece en el dispositivo, mostrándose la pantalla principal tanto del X73PHDManager, por un lado, como del X73PHDAgent (ver Figuras 3.6a y 3.8a). Pulsando sobre el botón **Start** del manager, se pasa a la pantalla donde se recibirán los datos de las medidas enviadas por el agente que, lógicamente, aparecerá vacía inicialmente (ver Figura 3.6b). En este momento, en el manager se ha abierto un *socket* y éste se encuentra "escuchando" las conexiones entrantes que le puedan llegar del agente (ver Figura 4.8).

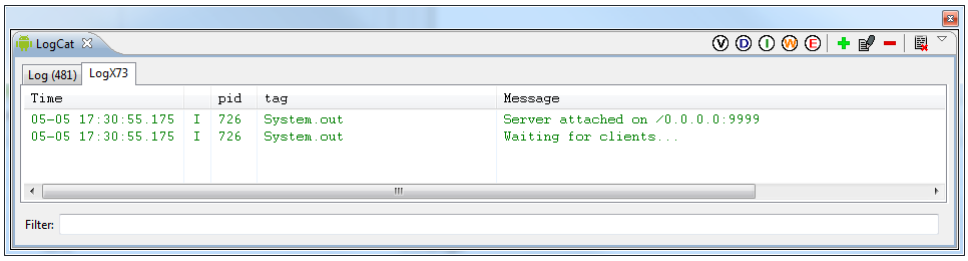


Figura 4.8 – Log del manager en la consola de Eclipse

En el agente, la primera operación será pulsar el botón **Scan** para buscar los dispositivos Bluetooth disponibles a su alcance, y conectarse con el correspondiente al manager. Se mostrará una pantalla con los equipos previamente conocidos, y la opción de realizar un nuevo escaneado (ver Figura 4.9a). Seleccionando uno de ellos, ambos terminales se conectan y la interfaz vuelve a su pantalla original, pero mostrando ahora un mensaje de "CONNECTED" en la parte superior (ver Figura 4.9b).

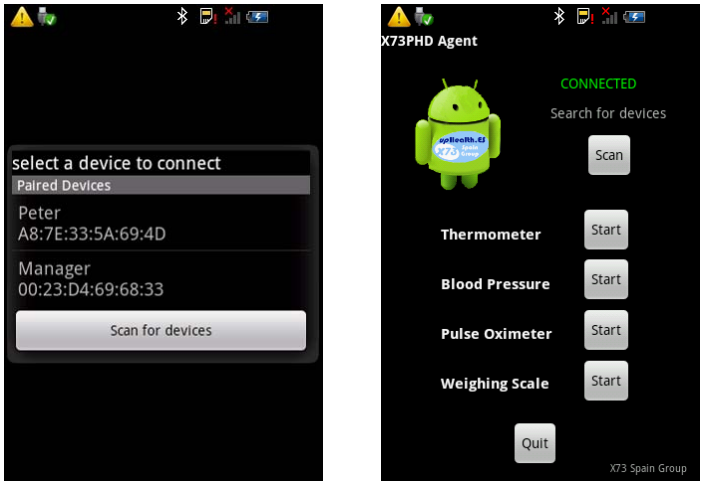


Figura 4.9 – a) Pantalla del Scan y b) Bluetooth conectado

Una vez que agente y manager están conectados a través de Bluetooth, pulsando en el botón **Start** del MD que se quiera utilizar como agente, aparece la pantalla correspondiente a dicho dispositivo médico (ver Figura 3.8b). Inicialmente, se puede ver que el estado de la FSM se encuentra en *Unassociated*. Para llevar a cabo la conexión, se deberá pulsar el botón que en este momento aparece como **Disconnected**, iniciándose de esta forma el proceso de asociación (*Associating*), es decir, se enviará el *AssociationRequest*, y según sea el *AssociationResponse* enviado por el manager, se pasará, bien al estado *Configuring* (si la configuración del agente no se conocía previamente) o bien al estado de *Operating* (ver Figura 4.10).

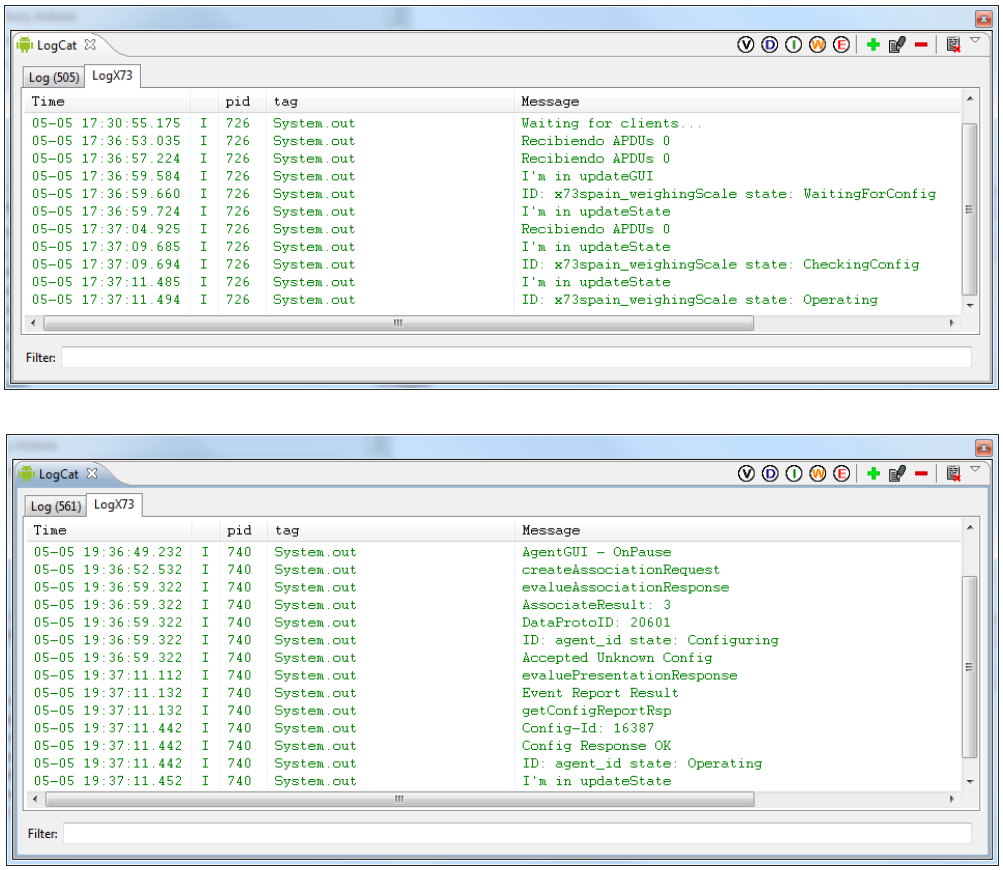


Figura 4.10 –Proceso de asociación a)del manager y b)del agente

Siempre que la conexión haya sido satisfactoria, ambos terminales mostrarán en sus pantallas el estado *Operating*, y las luces que antes eran rojas, ahora serán de color verde; además, en el agente el botón que aparecía desconectado ahora se encenderá y se mostrará como *Connected* (ver Figura 4.11).

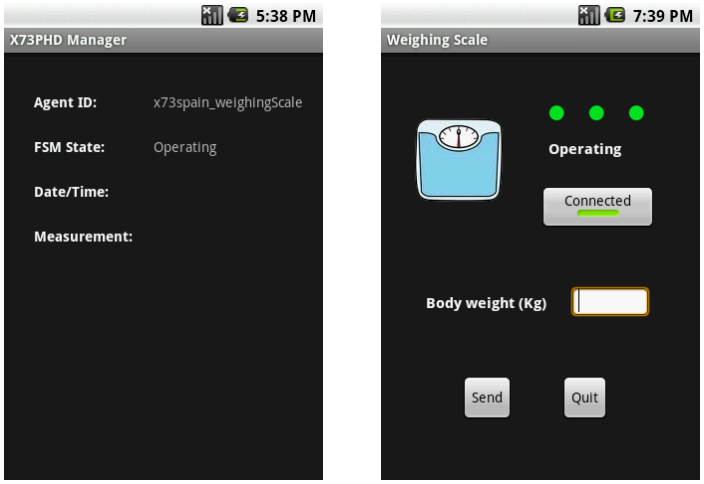


Figura 4.11 – Manager y Agente en modo Operating

Una vez que ambos se encuentran asociados correctamente en modo *Operating*, se procede al envío de la medida. Para ello, basta con introducir el valor correspondiente en la casilla habilitada para ello, en las unidades adecuadas, y pulsar el botón **Send**. Si el dato introducido contiene decimales y la medida en cuestión no los soporta, aparecerá un mensaje en la pantalla advirtiendo del conflicto, de la misma forma que si los decimales están habilitados pero se han escrito de manera incorrecta, con una coma en lugar de un punto, por ejemplo (ver Figura 4.12).

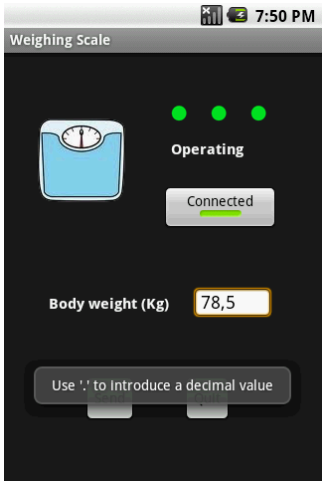


Figura 4.12 – Mensaje de advertencia sobre el dato introducido

Después de pulsar el botón **Send**, y habiendo introducido la medida correctamente, se inicia el proceso de comunicación de datos, por el cual se envían tanto la medida como la fecha y la hora en la que ha sido tomada, que no será otra que la del agente utilizado. Si el manager responde que los datos han sido recibidos de forma correcta, el agente mostrará un mensaje de confirmación al usuario. A su vez, la pantalla del manager se actualizará para representar los datos de la medida recibida (ver Figura 4.13).

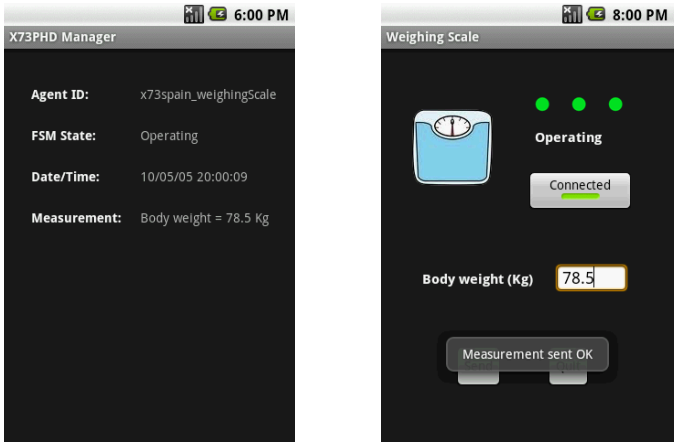


Figura 4.13 – Envío y recepción del dato médico a)Manager b)Agente

Para finalizar el proceso existen dos opciones, pulsar el botón que ahora aparece como *Connected*, o bien pulsar el botón **Quit**. En ambos casos, se procederá a la desasociación de agente y manager de la misma forma: se enviará una petición de *ReleaseRequest* que, si es aceptada por el manager, dará lugar a la desconexión de ambos, pasando al estado *Disconnected*.

Una vez finalizado el proceso, el manager seguirá esperando posibles conexiones futuras, si bien se actualizará la pantalla eliminando la medida recibida anteriormente y mostrando el estado *Disconnected*. Al mismo tiempo, el agente volverá a su pantalla principal (ver Figura 3.8a) si se ha pulsado el botón **Quit**, donde se podrá elegir un nuevo dispositivo (o incluso el mismo) y repetir todo el procedimiento de nuevo; o permanecerá en la misma pantalla si simplemente se ha desconectado a través del conmutador *Connected/Disconnected* (ver Figura 4.14).

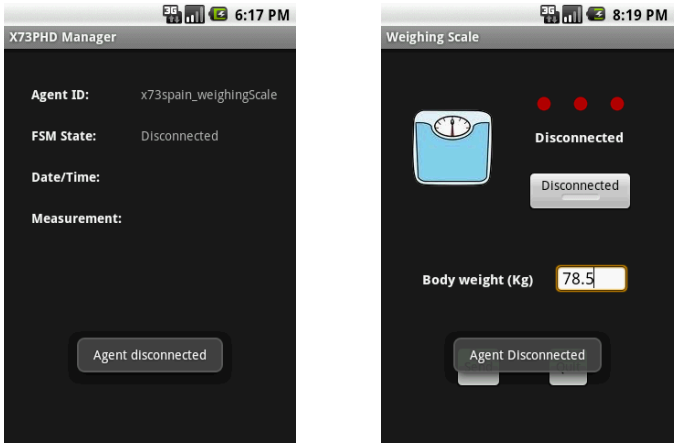


Figura 4.14 – Desconexión a)del manager y b)del agente

4.3.1 Sistema de ficheros XML

Otra de las funcionalidades implementadas, es la de guardar los datos recibidos por el manager en un fichero XML. Para ello, se debe pulsar el botón **MENU** del terminal, y automáticamente aparecerá un menú contextual en la parte inferior de la pantalla (ver Figura 4.15a), ofreciendo tres opciones: salir de la pantalla de recepción de datos volviendo a la interfaz principal (**Quit**); guardar la medida en el archivo mencionado (**Save**); y enviar dicha medida a un servidor de HCE (**Send**).

Pulsando el botón **Save** los datos se guardarán y el dispositivo preguntará, a través de un *pop-up*, si se desean guardar más medidas (ver Figura 4.15b). Esto es útil si se quieren almacenar varias medidas en el mismo archivo o si, por el contrario, se prefiere tener cada medida por separado en ficheros diferentes.

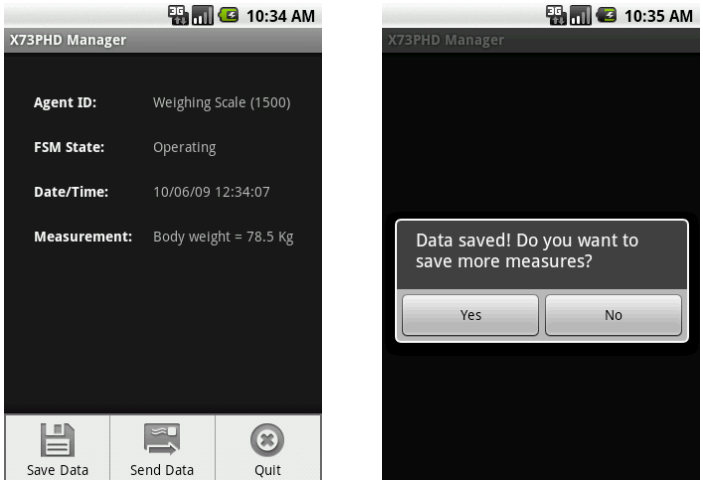


Figura 4.15 – a)Menú contextual para guardar la medida o salir, b)Pop-up para guardar varias medidas en un fichero

Seleccionando la opción **No**, el proceso de almacenamiento finalizará y el fichero en cuestión aparecerá en una carpeta con el nombre "x73spain", dentro de la tarjeta de memoria del terminal (*SDcard*). El nombre de éste estará formado por el nombre del agente que envió los datos y por la fecha (ver Figura 4.16). En caso de haber más de un archivo del mismo dispositivo médico y con la misma fecha, se añadirá un número al final del nombre para distinguirlos. Si por el contrario se elige la opción **Yes**, el fichero permanecerá abierto mientras se envían otras medidas para que, de esta forma, todas se encuentren dentro del mismo archivo XML.

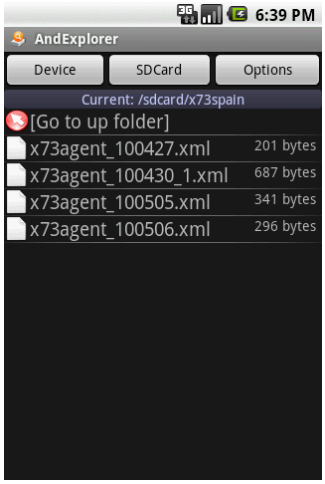


Figura 4.16 – Ficheros XML guardados en la carpeta x73spain

Por último, si se elige la opción **Send**, el fichero XML generado se envía a un servidor de HCE (*EHR Server*) a través de *Web Services*, conectándose para ello bien a un enlace WiFi

disponible, o bien a través de una conexión 3G. Si el envío se ha realizado de forma correcta, aparecerá un mensaje en la pantalla con la indicación correspondiente (ver Figura 4.17).

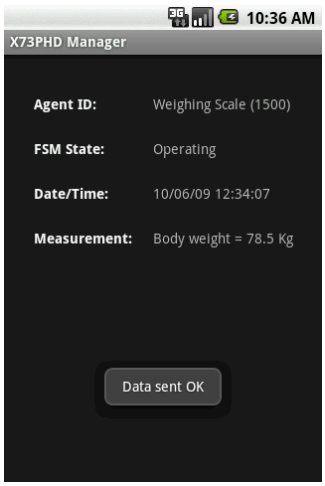


Figura 4.17 – Mensaje de confirmación de envío del XML

El servidor HCE forma parte una de las líneas de trabajo del grupo X73Spain de la UZ. Por lo tanto, el formato del XML y la comunicación entre el manager HTC/Android y el servidor en cuestión, se ha desarrollado según lo establecido junto con las personas responsables de la investigación llevada a cabo sobre el HCE, tal y como dicta la norma EN13606. La integración con este servidor, permite disponer de una plataforma completa (*end-to-end*), dando soporte desde el inicio del proceso (toma de la medida por el paciente) hasta la parte final (almacenamiento de la medida en el HCE).

5. Evaluación y resultados

En este apartado se explican las pruebas realizadas para comprobar que el proyecto funciona correctamente en escenarios de varios tipos. Se muestran además las tramas de datos enviadas entre agente y manager, viendo cómo siguen el comportamiento establecido por la norma X73. Se tratan además las diferentes pruebas de interoperabilidad e integración con otros entornos y plataformas. Por último se exponen los resultados obtenidos y los problemas encontrados.

5.1 Prueba de software

Para llevar a cabo las pruebas y comprobar que el código desarrollado efectúa correctamente las operaciones para las cuales has sido diseñado, se utilizará el emulador que proporciona el *plugin* de Android para Eclipse, a través de su herramienta AVD (*Android Virtual Device*). Esta utilidad permite crear de forma rápida y sencilla unidades virtuales que simularán los terminales reales. En esta primera fase de pruebas, ambos dispositivos (agente y manager) se encontrarán en el mismo PC; se tendrán, por tanto, dos instancias del emulador sobre el mismo entorno Eclipse, trabajando en *localhost* y comunicándose a través de TCP.

El procedimiento a seguir para hacer funcionar el proyecto es el siguiente. Una vez creadas las dos unidades virtuales que emularán tanto agente como manager, faltará únicamente instalar los dos proyectos, uno en cada emulador, y ejecutarlos. Antes de llevar a cabo la comunicación entre ambos, es necesario realizar una última operación; debido a que los mensajes que envía el agente se dirigen a un puerto concreto del manager, habrá que abrir un terminal *telnet* en el manager para redirigir todas las conexiones entrantes hacia el puerto especificado en el agente. Para ello, basta abrir un terminal de consola en Windows y escribir la siguiente instrucción:

```
telnet localhost 5554
```

El número 5554, se refiere al puerto en el que se encuentra escuchando el emulador del manager (éste puede verse en la parte superior izquierda de la ventana donde se abre el emulador, y podría ser perfectamente otro número). Una vez ejecutado, se abrirá automáticamente el terminal telnet, donde se deberá escribir la instrucción de redirección, como puede verse en la Figura 5.1. En este caso, todas las conexiones que lleguen al manager, lo harán a través del puerto **9999**, que es el puerto contra el que se crea el *Socket TCP* en el agente, y en el que escuchará, por tanto, el manager.

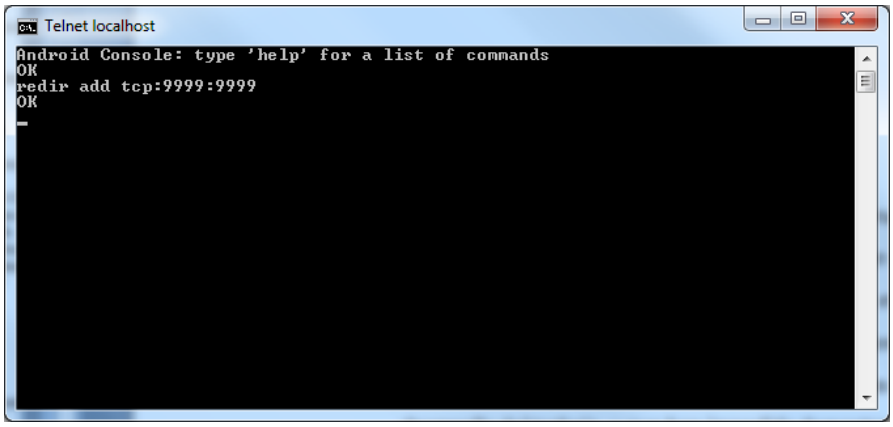


Figura 5.1 – Terminal telnet para redirigir puertos

Una vez ejecutada la redirección, ambas aplicaciones se encuentran listas para comenzar la comunicación. Después de haber visto en el punto 3.4 el funcionamiento del proyecto a nivel de usuario, la explicación se centra ahora en demostrar, a más bajo nivel, la correcta implementación del estándar X73, comprobando en todo momento las tramas de datos enviadas entre agente y manager y las transiciones entre los diferentes estados de la FSM definida por el estándar.

El primer paso consiste en inicializar el manager pulsando el botón **Start**. De esta forma, se abrirá el socket mencionado en el puerto 9999, esperando las conexiones entrantes que le puedan llegar desde el agente. En este momento, ambos dispositivos se encuentran en estado *Unassociated*. A partir de aquí, comenzará el envío y recepción de tramas, empezando, como se vio en el capítulo 3.1, por una petición de asociación del agente al manager (***AssociationRequest***), cuya trama de bytes puede verse en la imagen siguiente, separada según los atributos enviados (ver Figura 5.2). Se pasará entonces al estado de *Associating*.

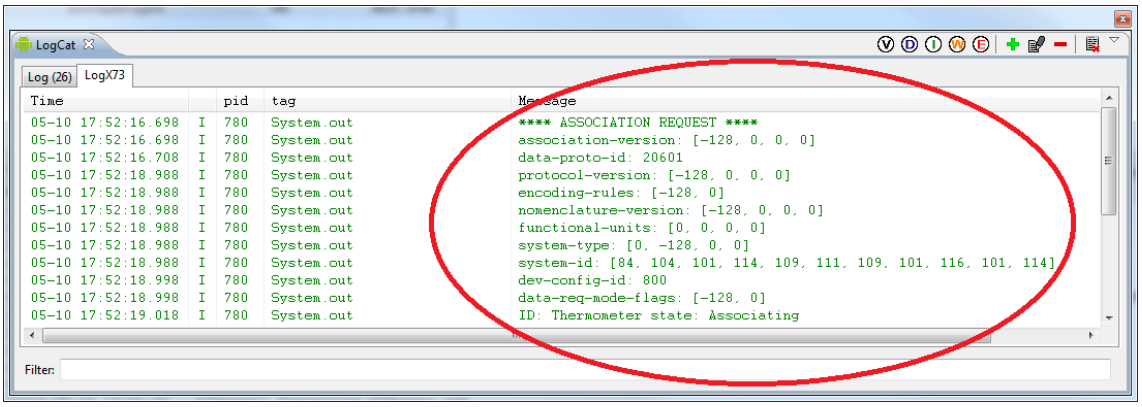


Figura 5.2 – Tramas de datos del AssociationRequest

A continuación, el manager responderá a la petición de asociación con un mensaje del tipo ***AssociationResponse***, en el cuál informará de si la petición ha sido aceptada, rechazada, o si necesita más datos de configuración para aceptarla, algo que ocurrirá siempre que el agente no haya sido registrado previamente en el manager, es decir, que no se hayan asociado en anteriores ocasiones. Para saber cuál ha sido la respuesta del manager, éste envía un código de asociación (*AssociateResult*), que permitirá distinguir al agente entre los diferentes casos. Estos códigos son:

- **0: Association Request Accepted**
- 1: Association Request Rejected Permanent
- 2: Association Request Rejected Transient
- **3: Association Request Accepted Unknown Config**
- 4: Association Request Rejected No Common Protocol
- 5: Association Request Rejected No Common Parameter
- 6: Association Request Rejected Unknown
- 7: Association Request Rejected Unauthorized
- 8: Association Request Unsupported Association Version

Según los códigos, si la respuesta tiene valor 0, la petición de asociación ha sido aceptada, mientras que si tiene valor 3, ésta ha sido aceptada pero sin conocer la configuración del agente, por lo que el agente deberá enviar un ***ConfigReport*** con sus parámetros de configuración, pasando al estado *Configuring* (ver Figura 5.3). Para el resto de códigos, la petición es rechazada y no se dará lugar a la asociación.

Una vez enviada la configuración, el agente esperará la confirmación de ésta por parte del manager, que vendrá dada, al igual que ocurría con la petición de asociación, por un código denominado ***ConfigResponse***, el cual puede tomar los siguientes valores:

- **0: Config Response OK**
- 1: Unsupported Config
- 2: Standard Config Unknown

Así pues, si este código tiene valor 0, el manager habrá aceptado la configuración del agente y ambos pasarán al estado *Operating*. Llegados a este punto, ambos dispositivos se encuentran ya asociados y se podrá iniciar la transferencia de las medidas tomadas por el agente.

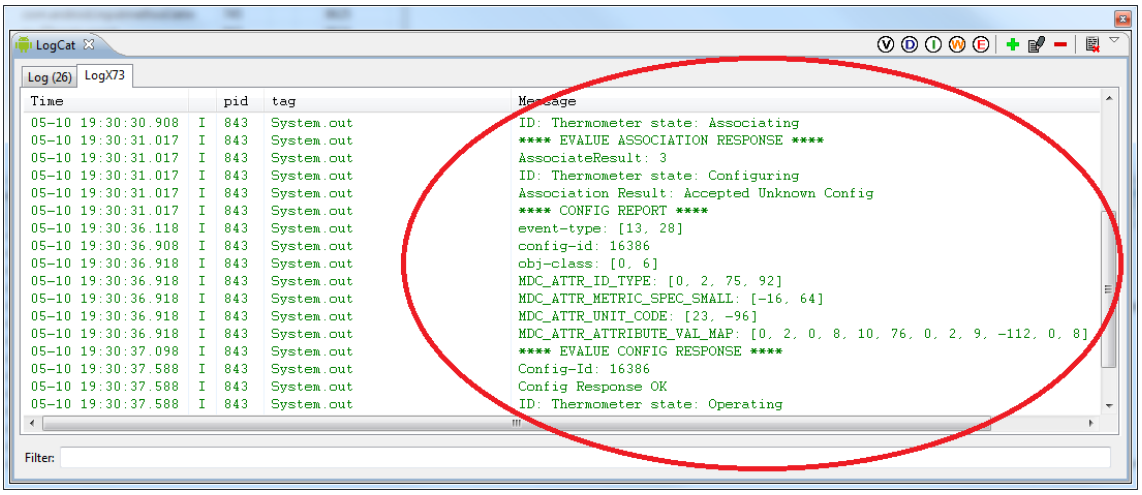


Figura 5.3 – Tramas de datos del ConfigReport

El agente mandará entonces un ***DataReport*** que contendrá los datos de la medida (el dato médico y la fecha y la hora) y esperará, al igual que en casos anteriores, a que el manager confirme la recepción de los datos enviados. En la Figura 5.4, es importante destacar la línea que aparece remarcada (*obs-scan-fixed*), pues es la que contiene la información; los dos primeros bytes corresponden al propio dato clínico, mientras que el resto son los correspondientes a la fecha y la hora en la que éste ha sido tomado.

Ahora se podrían seguir enviando nuevas medidas siguiendo el mismo proceso, es decir, creando nuevos mensajes *DataReport* con los datos clínicos; o bien finalizar la comunicación a través de una petición de desasociación por parte del agente (***AssociationReleaseRequest***), y de la consiguiente confirmación del manager (***AssociationReleaseResponse***). Para ello, el agente únicamente deberá indicar el motivo de la

finalización, que será '0' para la desconexión normal (*Normal Reason*), mientras que el manager responderá con el mismo código para indicar que acepta la petición de desasociación.

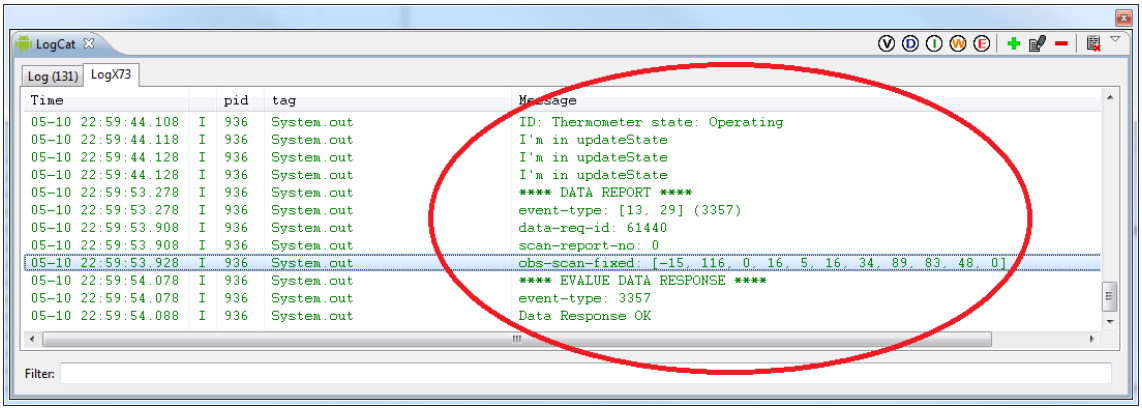


Figura 5.4 – Tramas de datos del DataReport

5.2 Pruebas de interoperabilidad

Después de haber comprobado que el *software* desarrollado funciona correctamente cuando agente y manager se encuentran en el mismo equipo (ver Figura 5.5a), se procederá a probar el proyecto en diferentes escenarios.

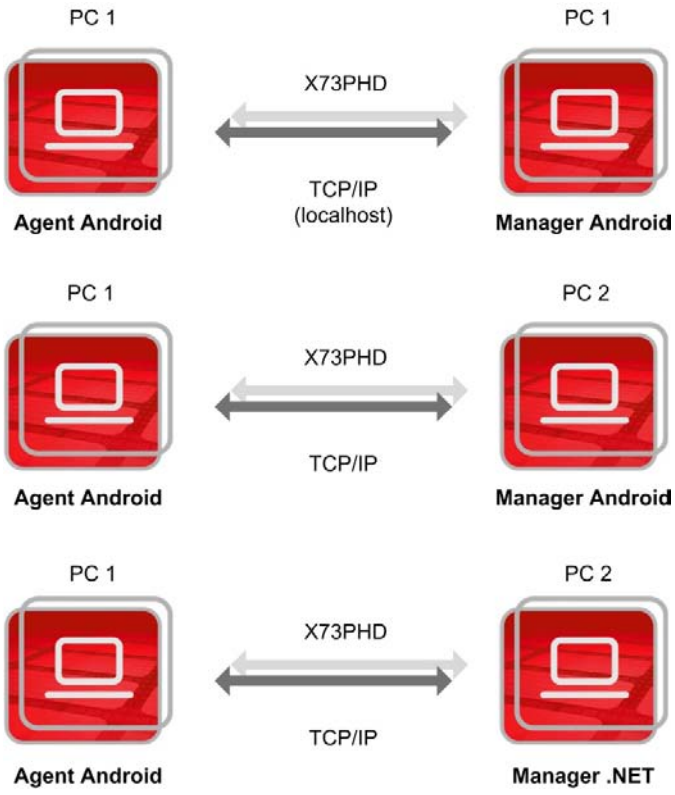


Figura 5.5 – Escenarios de pruebas de interoperabilidad

Para ello, se pasará de un entorno *localhost* (agente y manager en el mismo PC) a otro con dos PCs. Ambos equipos seguirán conectados por IP a través de un hub o switch, por lo que la comunicación se realizará mediante TCP (ver Figura 5.5b). Ya que en las dos situaciones se trata de los mismos agente y manager sobre Android (emulados con Eclipse), el funcionamiento será exactamente el mismo que el explicado a lo largo de esta memoria, salvo que ahora, las peticiones del agente deberán dirigirse a la dirección IP del manager, y no a *localhost* como se hacía durante las primeras pruebas.

Un reto importante llegado este punto, consiste en comprobar que la plataforma desarrollada es absolutamente compatible con otros entornos de desarrollo. Aprovechando que en el grupo X73Spain se está trabajando también en otra plataforma de telemedicina, equivalente a la que ocupa este proyecto, sobre tecnología .NET, una prueba importante sería comunicar las dos implementaciones. De esta forma, se procede a la comunicación de un agente Android sobre PC, con un manager .NET funcionando en otro PC distinto (ver Figura 5.5c).

Como es lógico, el funcionamiento del agente Android será siempre el mismo, especificando la dirección IP del manager .NET al igual que en el caso anterior. Como ambos desarrollos están diseñados cumpliendo con las pautas establecidas por la norma X73, la comunicación se realiza sin ningún problema y los datos médicos son transmitidos correctamente (ver Figura 5.6).

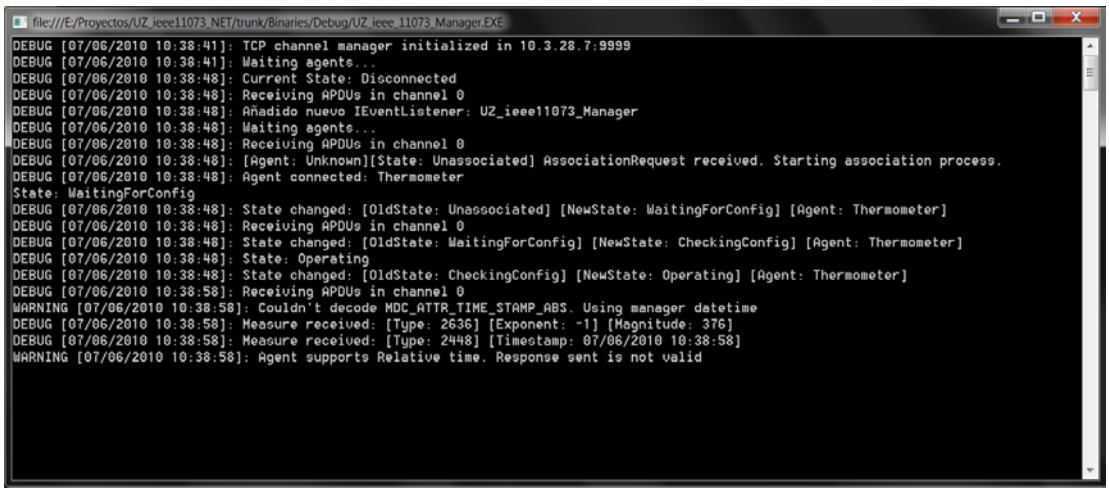


Figura 5.6 – Captura manager plataforma .NET

5.3 Pruebas de hardware

Como ya se detalló anteriormente en el capítulo 3.1, al no disponer de dispositivos médicos reales que cumplan con la norma, ha sido imposible realizar las pruebas pertinentes en un escenario de trabajo real, por lo que se ha trabajado en todo momento con MDs simulados sobre un terminal Android.

A pesar de todo, está previsto adquirir en las próximas semanas un dispositivo médico certificado por Continua para realizar este tipo de pruebas, y poder trasladar el trabajo a un entorno clínico real, como puede ser un centro médico, una residencia de ancianos, etc.

Tanto en el punto 5.1 como el 5.2, las pruebas se han llevado a cabo mediante terminales emulados en un PC, a través de la herramienta que ofrece Eclipse para ello. Sin embargo, una prueba con los terminales reales a través de Bluetooth es completamente necesaria, ya que éste será el modo de trabajo de la plataforma. Por lo tanto, se ha instalado en los móviles HTC disponibles el X73PHDAgent y el X73PHDManager, y se ha repetido el proceso de conexión, asociación y envío de datos médicos, llevándose a cabo, como estaba previsto, de forma satisfactoria.

Por último, otra prueba realizada ha sido la concerniente al sistema de ficheros XML. Al igual que ocurría con los emuladores, el fichero se crea correctamente y se almacena en el terminal móvil. Pero sin duda, el punto de mayor interés es el relativo al envío de este XML al servidor de HCE.

Para ello, a través de un punto de conexión WiFi configurado en el laboratorio, el manager se conecta a la red y envía, a través de HTTP, el fichero en cuestión, recibiéndose en el servidor y finalizando así el proceso completo de la toma y transmisión de los datos médicos. La dirección donde se encuentra ubicado este servidor es:

`http://uzserver.no-ip.org/CE2EHR/`

6. Cronograma de implantación

En este capítulo se explica cuál ha sido la cronología seguida durante la realización del PFC, y cómo se han desarrollado las diferentes fases que lo componen, representando para ello el diagrama de Gantt.

6.1 Cronograma de implantación

A continuación se expone el tiempo de dedicación que ha sido empleado, aproximadamente, para cada una de las tareas que componen este proyecto.

Las primeras semanas de trabajo se analizó la norma X73 y algunos documentos escritos por el grupo X73Spain de la Universidad de Zaragoza.

Con una idea global sobre lo definido por la norma, se pasó a estudiar la anterior plataforma de telemonitorización que trabajaba sobre Windows Mobile (plataforma 2.1 – BT) [11], la cual se debía migrar al nuevo sistema operativo Android. Así se comprendió con mayor detalle cómo se implementaban las especificaciones definidas en el estándar.

Durante este estudio, se descubrió la existencia de un proyecto que podría ser útil para la realización de este PFC, el proyecto OpenHealth desarrollado por la Universidad Rey Juan Carlos de Madrid a través de su programa Morfeo. Por lo tanto, se comenzó a analizar, llegando a la conclusión de que sería más conveniente partir de este nuevo diseño, que migrar la plataforma anterior al completo. Después de una reunión con los responsables de OpenHealth, se decidió utilizar su desarrollo como punto de partida del PFC. Además, y derivado de esta reunión, una nueva tarea llevada a cabo durante el transcurso del PFC ha sido la de mantener una línea de contacto con el grupo de Morfeo.

Al mismo tiempo que se estudiaba el trabajo realizado por los compañeros de la URJC, se comenzó la familiarización con el entorno Android, haciendo pequeños programas e implementaciones guiadas para aprender a crear aplicaciones con Eclipse para este sistema operativo. A su vez, se tuvo que realizar un breve tutorial de Java, concretamente de J2ME, recordando los conceptos básicos de la programación en este lenguaje.

Una vez asimilado el funcionamiento del proyecto OpenHealth, se afrontó el diseño y arquitectura a implementar en este PFC, comenzando con la etapa de programación del código, que supone la tarea central del proyecto y ocupa la mayor parte del trabajo realizado. Cuando se había avanzado suficiente con el desarrollo del código, se comenzaron a realizar pruebas para comprobar que todo funcionaba según lo previsto, modificando pequeños detalles y añadiendo nuevos aspectos.

Esta fase de pruebas y mejoras, junto con la redacción de la presente memoria, se extiende hasta la fecha de finalización y entrega del proyecto.

6.2 Diagrama de Gantt

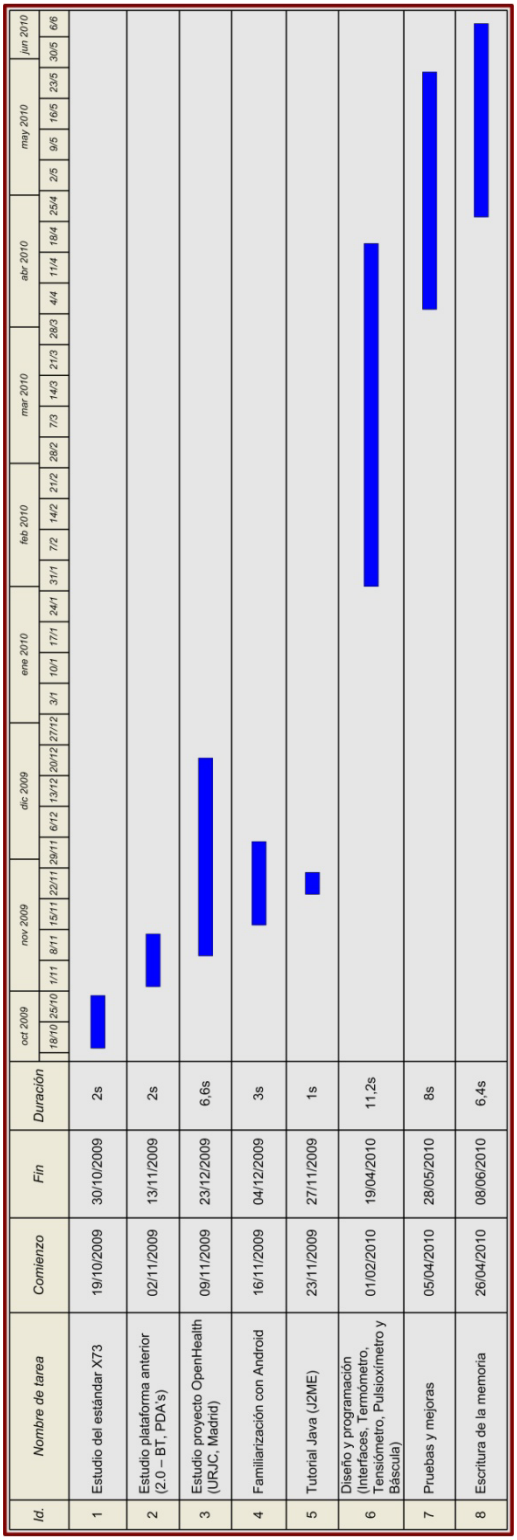


Figura 6.1 – Diagrama de Gantt

7. Conclusiones y líneas futuras

En este apartado se ofrece una visión general a posteriori de lo que se ha hecho en este PFC, comparándolo con la idea original, y reflejando las conclusiones extraídas de la realización del proyecto. Se plantean además una serie de líneas futuras que podrían seguirse para continuar y mejorar el trabajo realizado.

7.1 Conclusiones

Este proyecto plasma la idea de abrir la plataforma de telemonitorización existente a nuevos entornos *open source*, creando así una nueva plataforma basada en Android. Consigue además implementar nuevos dispositivos médicos a parte del tensiómetro que se había desarrollado en la anterior versión 2.1 – BT, apareciendo de esta forma el termómetro, el pulsioxímetro y la báscula.

Por otro lado, y aprovechando la facilidad que ofrece para ello el nuevo sistema operativo Android, se han creado unas interfaces de usuario mucho más amigables, mejorando así la usabilidad de la plataforma, ya que son vistosas y fáciles de utilizar.

El hecho de tratarse de una plataforma *open source* ofrece muchas potencialidades, ya que permite su completa integración con otros desarrollos, como puede ser el trabajo realizado por los compañeros de Morfeo, además de la integración en entornos multiplataforma, y el acceso a las capas nativas, como es el caso de la pila Bluetooth de Android o Linux, algo que no está permitido en sistemas propietarios como Windows.

Con la utilización de este PFC, se aumentaría notablemente la calidad de vida de muchos pacientes, sobre todo aquellos de avanzada edad que deben acudir de forma periódica y continuada a los centros de salud a tomarse medidas rutinarias que, de esta forma, podrían tomarse en su propia casa. Además, como efecto secundario, se produciría un ahorro importante en el sector sanitario, tanto en tiempo como, indirectamente, en dinero.

Con respecto al estándar, se ha llegado a la conclusión de que es una pieza fundamental en el desarrollo de este tipo de sistemas de telemedicina. Gracias a su implantación, la interoperabilidad entre equipos de diferentes marcas y modelos es absoluta, lo que dota de mayor potencial a los dispositivos que lo cumplen. De esta forma, se facilita el desarrollo de nuevos sistemas por parte de los fabricantes y los grupos de investigación, y se ofrece al usuario final la ventaja de poder comprar cualquier equipo, del tipo y fabricante que sea, siempre y cuando siga la norma X73.

Personalmente, he aprendido muchísimo trabajando en este proyecto, pues me atraía la idea de desarrollar una aplicación para Android ya que, desde mi punto de vista, tiene un altísimo potencial y va a ocupar una posición destacada tanto en el mercado como en el sector de la telemedicina. Además, al tratarse de una plataforma *open source* y estándar, supone una motivación extra, debido a que mi trabajo ha sido algo real, tangible, y que puede suponer un salto cualitativo importante en el ámbito de la medicina, y servir además como punto de partida de posibles desarrollos futuros en este campo.

7.2 Líneas futuras

Como posibles mejoras de este proyecto, existen varios puntos sobre los que se podría actuar para hacer que la plataforma ofrezca nuevas opciones y funcionalidades. La mejora más inmediata es la de poner en funcionamiento la plataforma con dispositivos médicos reales. Para ello es necesario adquirir equipos médicos que cuenten con tecnología Bluetooth y que estén certificados por Continua, es decir, que cumplan con la norma X73.

Por otra parte, dejando de lado esta serie de pruebas en entornos reales, existen también varias ampliaciones del proyecto que lo dotarían de mayores posibilidades. Una de ellas, es la de implementar un sistema de alarmas a través del cual, el personal médico pudiera avisar de alguna forma al paciente, sobre cuándo se debe tomar y enviar una medida. Para conseguir esto, y siendo que esta plataforma se basa en el sistema operativo Android, perteneciente a Google, una opción sencilla sería utilizar el servicio gratuito Google Calendar. Éste permite la creación de calendarios compartidos, de tal forma que el doctor podría crear eventos/citas en el calendario del paciente, apareciendo las alarmas correspondientes en la pantalla del terminal en el momento oportuno.

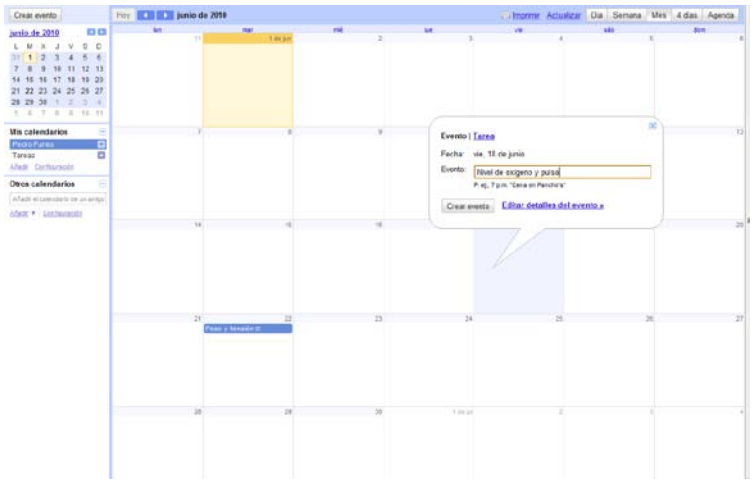


Figura 7.1 – Gestión de alarmas con Google Calendar

En esta misma dirección, se está realizando una tesis fin de máster que estudia la política de notificación de alarmas. Según este trabajo, las notificaciones (*reminders*) podrán ser de dos tipos: visuales o sonoras; dependiendo de posibles discapacidades, sobre todo en personas mayores, donde los avisos sonoros son muy importantes. Un aspecto a destacar es que todos los eventos serán tratados por la aplicación, pero solo algunos de ellos se mostrarán como alarmas al paciente, ya que no es necesario notificar ciertos eventos de bajo nivel.

En la tesis comentada, se aborda además el concepto de "accesibilidad" y "usabilidad" dentro del ámbito médico, para que cualquier paciente/usuario, con o sin diversidad funcional, pueda hacer uso de ella, lo que le proporcionaría mayor independencia, y le ofrecería la posibilidad de prescindir de ayuda asistencial para tareas rutinarias como pueden ser la toma de la temperatura, la tensión o el pulso. Para ello, sería necesario diseñar una interfaz accesible, usable e interoperable, además de multi-plataforma, para que se pueda instalar en distintos entornos como PDAs, PCs, Tablet PC… e incluso dejar abierta la posibilidad de una futura instalación en televisión, a través de un canal específico de telemedicina.

Siguiendo con los servicios a usuarios, otra línea futura sería crear interfaces personalizadas para cada usuario. Esto sería muy útil, por poner un ejemplo, en una pequeña residencia de ancianos donde sólo se dispone de un dispositivo médico de cada tipo, y con él se toman las medidas de todos los residentes. Aplicando este sistema de interfaces, en el teléfono móvil del personal sanitario se podrían asociar las medidas recibidas a cada uno de los pacientes, para así tener un registro personalizado de su estado de salud, ya que cada paciente tendría asignado una serie de dispositivos médicos, pues no todos necesitan tomarse las mismas medidas.

Referencias

[1] California Telehealth & Telemedicine Center - CTTC
http://www.itelemedicina.com (Última visita a la página web: Mayo de 2010)

[2] R. Wooton, J. Craig, "Introduction to Telemedicine", ISBN-10: 1853156779, *Rittenhouse Book Distributors*, 2nd Edition, 2006.

[3] J. Korhonen, M. Parkka, "Health Monitoring in the Home of the Future: Wear it well", *IEEE Eng Med Biol Mag*, 22(3): 66-73, 2003.

[4] S. Pedersen and W. Hasselbring, "Interoperability for information systems among the health service providers based on medical standards," *Informatik - Forschung Und Entwicklung* 18(3-4):174-188, 2004.

[5] C.E. Chronaki and F. Chiarugi, "Interoperability as a Quality Label for portable & wearable Health Monitoring Systems", *Personalized Health: The Integration of Innovative Sensing, Textile, Information & Communication Technologies, Studies in Health Technology and Informatics Book Series, IOS Press*, 2005.

[6] ISO/IEEE11073. Health informatics. Point-of-Care Medical Device communication (x73PoC-MD)
http://www.ieee1073.org First edition: 2004 (Última visita a la página web: Mayo de 2010)

[7] ISO/IEEE11073. Health informatics. Personal Health Devices communication (X73PHD)
[P11073-00103. Technical report - Overview]
[P11073-104xx. Device specializations]
[P11073-20601. Application profile - Optimized exchange protocol]
http://standards.ieee.org/ First edition: 2006 (Última visita a la página web: Mayo de 2010)

[8] Open Movil Forum (OMF)
http://open.movilforum.com (Última visita a la página web: Abril de 2010)

[9] Open Handset Alliance
http://www.openhandsetalliance.com/ (Última visita a la página web: Mayo de 2010)

[10] X73Spain Group
[09X73SG Informe 09sem2]
http://www.x73spain.com (Última visita a la página web: Mayo de 2010)

[11] Pilar del Valle. Proyecto Fin de Carrera. "Implementación de la plataforma 2.0 para telemonitorización ubicua de pacientes basada en el protocolo X73-PHD sobre Windows Mobile en dispositivo PDA".

[12] Morfeo OpenHealth (Grupo GSyC de la Universidad Rey Juan Carlos de Madrid)
http://openhealth.morfeo-project.org/?lng=es (Última visita a la página web: Mayo de 2010)

[13] Android Market – Development phones
http://market.android.com/ (Última visita a la página web: Mayo de 2010)

[14] Integrating the Healthcare Enterprise (IHE)
http://www.ihe.net/ (Última visita a la página web: Mayo de 2010)

[15] Continua Health Alliance
http://www.continuaalliance.org (Última visita a la página web: Mayo de 2010)

[16] Telequia – Android también en la telemedicina
http://www.telequia.es (Última visita a la página web: Mayo de 2010)

[17] Linux Devices – Android app offers remote clinical diagnostics
http://www.linuxfordevices.com (Última visita a la página web: Mayo de 2010)

[18] NONIN
http://www.nonin.com/ (Última visita a la página web: Mayo de 2010)

[19] OMRON
http://www.omron.com/ (Última visita a la página web: Mayo de 2010)

[20] I. Martínez, J. Escayola, J.D.Trigo, M. Martínez-Espronceda, L. Serrano, P. Muñoz, J. García. "Plataforma Telemática de Integración de Estándares End-to-End para Salud Personal". *Jornadas de Ingeniería Telemática.* 2009

[21] Binary Notes
http://bnotes.sourceforge.net/ (Última visita a la página web: Abril de 2010)

[22] ITU – Introduction to ASN.1
http://www.itu.int (Última visita a la página web: Abril de 2010)

[23] Xataka Movil
http://www.xatakamovil.com (Última visita a la página web: Mayo de 2010)

[24] ANDROIDSIS – Comparativa terminales
http://androidsis.com (Última visita a la página web: Mayo de 2010)

[25] Android Developers
http://developer.android.com (Última visita a la página web: Mayo de 2010)

[26] Mitjans Galito, J. y Esteve Reig, J. "Enfermería. Técnicas clínicas". Ed. McGraw-Hill Interamericana. Primera Edición. Año 2002.

[27] Tortora, G.J. y Grabowski, S.R. "El aparato cardiovascular: el corazón". Ed. Harcourt Brace. Año 1998.

[28] MedlinePlus
www.nlm.nih.gov/medlineplus (Última visita a la página web: Mayo de 2010)

[29] Tu otro médico
www.tuotromedico.com (Última visita a la página web: Mayo de 2010)

[30] OMS – Organización Mundial de la Salud
www.who.int (Última visita a la página web: Mayo de 2010)

Anexo A – La norma ISO/IEEE 11073

En la actualidad, no existe ninguna norma que aborde oficialmente el problema de la integración de dispositivos en entornos de telemonitorización domiciliaria y ambulatoria, pero sí hay una familia de normas orientada a la interoperabilidad de dispositivos médicos en el punto de cuidado, que puede aplicarse en buena medida. Se trata de la familia ISO/IEEE 11073 (también nos referiremos a ella como X73), que está sufriendo modificaciones para cubrir ese campo.

La norma ISO/IEEE 11073, como ya se comentó en la introducción, es un conjunto único de normas desarrolladas y adoptadas por todos los países para la conectividad completa de dispositivos médicos, que aporta interoperabilidad, *plug&play*, transparencia y facilidad de uso y configuración. Dicha norma está, a fecha de finalización de este documento, en fase de desarrollo. Para su desarrollo se han unido las organizaciones CEN, IEEE e ISO, entre otras, uniendo esfuerzos anteriores que absorben normas previas como PoCMDC (interconexión de dispositivos e intercambio datos entre ellos) o las normas ENV13734 (VITAL) para las capas superiores y ENV13735 (INTERMED) para las capas intermedias.

Haciendo cronología, el IEEE fue el primero que desarrolló estándares en esta área con la aparición de Medical Information Bus (MIB) en 1984. Sin embargo, los principales fabricantes desarrollaron sus soluciones propietarias, que no han tenido una aceptación general. El CEN creó en 1993 un conjunto de estándares (*Point-of-Care Medical Device Communication*, PoC MDC) que fueron ratificados en 1999 para poder interconectar dispositivos e intercambiar datos. En el año 2000/2001 las organizaciones de estandarización IEEE e ISO se pusieron de acuerdo y crearon el "Pilot Project" para no competir y trabajar conjuntamente en una única serie de estándares. Apelando al tratado de Viena esta organización conjunta se extendió para incluir al CEN con el fin de llegar a una armonía internacional en los estándares. De hecho, estos acuerdos han puesto la base para que otras organizaciones de estandarización avancen de una forma similar y trabajen de manera coordinada con otras organizaciones como HL7, DICOM, IHE o Continua Alliance. A principios de 2004 se aprobaron los cinco estándares existentes de la norma 11073 y, desde entonces, se han desarrollado con un alto nivel de participación internacional. Están siendo adoptados como normas ISO a través de su comité técnico TC215, bajo la denominación de norma 11073. También se consideran estándares europeos por medio del TC251 del CEN.

Este proceso de integración comienza uniendo esfuerzos realizados anteriormente por cada una de las partes, de modo que se absorben normas previas de ISO e IEEE para poder llegar a cubrir todos los niveles/capas en la comunicación entre los dispositivos.

A partir de estas premisas, en los últimos años el vertiginoso desarrollo de nuevos MDs *wearables*, con sensores de alta calidad y sobre tecnologías *wireless* (como Bluetooth o ZigBee), y el incremento de accesos de banda ancha a redes multimedia, está acelerando la evolución del estándar X73 hacia una versión optimizada y adecuada a estas nuevas tecnologías y contextos ubicuos. Esta versión se denomina X73-PHD (*Personal Health Devices*). En esta ocasión cambia la arquitectura del protocolo, el modelo de comunicaciones MD-CE, se define una nueva máquina de estados finita (*Finite State Machine*, FSM), y el modelado del transporte y de nivel físico que conforman la pila de protocolos X73-PHD. Por todo ello, es evidente que este conjunto de estándares está todavía en fase de desarrollo, en un proceso evolutivo en el que multitud de ingenieros han trabajado en paralelo con universidades, instituciones y entidades internacionales. Esto desemboca en una situación transitoria en su progresiva implantación dado que no está asegurada todavía al no existir garantías de que los principales diseñadores y fabricantes de dispositivos médicos acepten el estándar. Sin embargo, la existencia de un estándar universal como X73-PoC para los sistemas sanitarios es una necesidad en la comunicación de los dispositivos médicos en e-Salud.

A la hora de diseñar las características del protocolo, es necesario evaluar los entornos de uso o *Use Cases* de los dispositivos candidatos a implementar X73-PHD de manera que quede convenientemente definido el rango de propiedades soportadas por el protocolo. En general el sistema posee una topología tipo estrella donde el CE situado en el centro recopila los datos provenientes de los agentes. Una lista preliminar de los casos contemplados para el desarrollo se encuentra en el borrador "IEEE-11073-00013. Draft Guide for Health informatics - Personal Health Device communication. Technical report – Overview". En él se destacan inicialmente los siguientes casos de uso:

- Salud y bienestar
- Fitness cardiovascular y monitorización de actividad
- Fitness de esfuerzo
- Gestión de enfermedades de alto riesgo: obesidad e hipertensión
- Assisted Ambient Living
- Cuidado de pacientes de avanzada edad

- Diabetes
- Monitorización de pacientes con problemas cardiacos

Durante el desarrollo de la norma, se han reagrupado en torno a tres tipos de uso como puede verse en Figura A.1. Aquí se incluyen además los dispositivos médicos asociados para su estandarización conforme a X73-PHD.

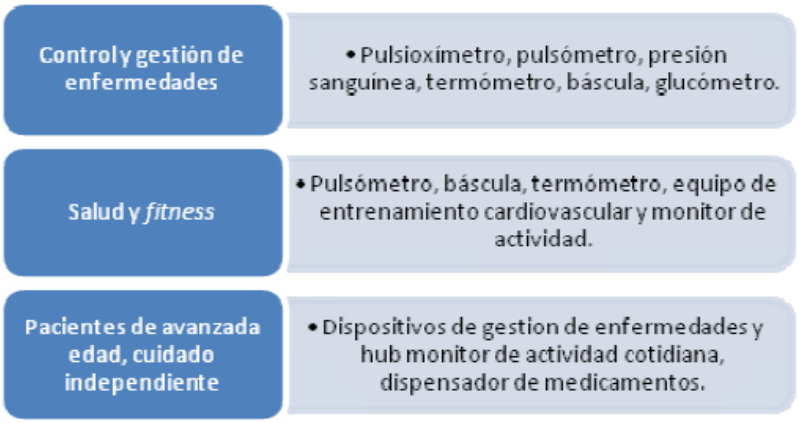


Figura A.1 – Tipos de uso para X73-PHD

A la hora de seleccionar el conjunto inicial de dispositivos que van a trabajar con el X73-PHD, se realizó una encuesta a diversos fabricantes en la cual se valora, para cada tipo de dispositivo del mercado, la necesidad de aplicar X73-PHD a sus comunicaciones. La clasificación final se hace en base al interés de los expertos en desarrollar un estándar para el dispositivo tanto a largo como a corto plazo, su posible participación en el desarrollo y evidentemente la existencia de un grupo de trabajo. Las especializaciones que se encuentran en proceso de desarrollo actualmente son las siguientes:

- 11073-10404 - Pulse oximeter
- 11073-10406 - Heart rate monitor
- 11073-10407 - Blood pressure monitor
- 11073-10408 - Thermometer
- 11073-10415 - Weighing scale
- 11073-10417 - Glucose meter
- 11073-10441 - Cardiovascular fitness and activity monitor
- 11073-10442 - Strength fitness equipment
- 11073-10471 - Independent living activity hub
- 11073-10472 - Medication Monitor

Esta lista inicial evoluciona y, conforme se obtienen versiones avanzadas de especializaciones en desarrollo, se van realizando nuevas propuestas que son llevadas a votación (*Project Approval Request*, PAR) para luego ser desarrolladas posteriormente si procede.

Las especializaciones de los dispositivos contienen a menudo definiciones de clases nuevas en el modelo de comunicaciones para abarcar las funcionalidades particulares del mismo. Además, el fabricante puede crear especializaciones propias o extendidas que contienen un esquema de objetos modificado, así como su lista de atributos definidos. Es tarea del CE el identificar estas configuraciones para decidir si acepta la asociación con el dispositivo en cuestión en base a sus capacidades.

Anexo B – Android

Android es una solución completa de *software* de código libre para teléfonos y dispositivos móviles. Es un paquete que engloba un sistema operativo, un *runtime* de ejecución basado en Java, una serie de librerías de bajo y medio nivel y un conjunto inicial de aplicaciones destinadas al usuario final. Se distribuye bajo una licencia Apache versión 2 (ASL2), una licencia libre permisiva que permite la integración con soluciones de código propietario.

Los primeros dispositivos con Android aparecieron en el último trimestre de 2008, aunque una versión preliminar del SDK, acompañado de un emulador y documentación de desarrollo, se encuentra disponible desde el 12 de noviembre de 2007 (ver Figura B.1).

Android buscaba (y lo ha conseguido) causar impacto en la industria de la comunicación móvil, estableciendo una plataforma abierta que permita un acceso fácil a prácticamente todas las funcionalidades *hardware* de los dispositivos en los que esté instalado, así como proveyendo de serie a los desarrolladores con librerías que favorezcan la creación ágil y rápida de aplicaciones. Se ha hecho especial énfasis en que las aplicaciones creadas por terceros no tendrán ningún tipo de desventaja en cuanto a funcionalidad y acceso al dispositivo frente a las aplicaciones "nativas" que se distribuyen originalmente con Android.

B.1 Arquitectura

Android proporciona un paquete completo de *software* a todos los niveles [4]:

- Un *kernel* Linux que sirve como base de la pila de *software* y se encarga de las funciones más básicas del sistema: gestión de drivers, seguridad, comunicaciones, etc.
- Una capa de librerías de bajo nivel en C y C++, como SQLite para persistencia de datos; SGL, desarrollada por Skia (adquisición de Google); OpenGL ES para gestión de gráficos 3D, con aceleración 3D opcional y Webkit como navegador web embebido y motor de HTML.

Figura B.1 – Evolución del lanzamiento de Android

- Un *framework* para el desarrollo de aplicaciones, dividido en subsistemas como el *package manager*, gestión del *hardware* del teléfono anfitrión (*telephony manager*) o acceso a APIs sofisticadas de geolocalización o mensajería XMPP. También incluye un sistema de vistas para manejar el interfaz de usuario de las aplicaciones, que ofrece posibilidad de visualización de mapas o renderizado HTML directamente en el interfaz gráfico de la aplicación.
- Una suite de aplicaciones (navegador, agenda, gestión del teléfono)

Las aplicaciones Android están programadas en Java, pero no corriendo sobre Java ME, sino sobre Dalvik, una máquina virtual Java desarrollada *ex profeso* por Google y optimizada para dispositivos empotrados y en la que los códigos fuente se compilan a ficheros de *bytecode* .dex. La creación de una *virtual machine* propia es un movimiento estratégico que permite a Google evitar conflictos con Sun por la licencia de la máquina virtual, así como asegurarse el poder innovar y modificar ésta sin tener que batallar dentro del *Java Community Process*.

B.2 Estructura de una aplicación Android

La estructura de una aplicación Android está definida por la interacción de distintos componentes, haciendo énfasis en la "agrupación débil" de distintas piezas. La aplicación hará uso de las distintas APIs expuestas por Android, de forma que los componentes encargados de realizar cada tarea puedan ser manipulados o reemplazados sin problemas, asegurando la máxima flexibilidad. Por ejemplo, una aplicación puede permitir al usuario elegir fotos mediante el componente "Galería" o, por ejemplo, reemplazar esa "Galería" por una selección de fotos a través de un servicio online. Los principales componentes de una aplicación serían:

- ✓ **Activity:** Representa cada una de las principales tareas que el usuario puede llevar a cabo en la aplicación. Típica (aunque no necesariamente) corresponderá a una pantalla específica de la aplicación y, también normalmente, una Activity será el punto de entrada (pantalla inicial) de nuestra aplicación. Desde ella se invocarán las vistas específicas o *layouts* para la aplicación.
- ✓ **IntentReceiver:** Permite a la aplicación declarar ciertos *callbacks* que responderán a cambios en el estado del terminal. Por ejemplo, llamada o email recibido, cambio en la geolocalización, etc.

- ✓ **Service:** Una tarea que corre en el *background* y que puede y debe ejecutarse sin interacción con el usuario. Una aplicación puede mandar los mensajes necesarios a un determinado servicio activo.
- ✓ **ContentProvider:** Establece una capa que permite a las distintas aplicaciones compartir datos. Con independencia del almacenamiento local que utilicen para sus propósitos, las aplicaciones necesitan declarar ContentProviders para poner a disposición de otros procesos los datos que consideren necesarios.

Estas son algunas de las principales, pero no las únicas piezas de construcción de la aplicación. También es interesante que se defina como pieza de primer nivel, el sistema de notificaciones en pantalla, que se recomienda como principal vía de comunicación con el usuario.

B.3 Dispositivos disponibles

Android ha conseguido "explotar" en el último año, por lo que desde que se inició este proyecto hasta su finalización, han aparecido numerosos dispositivos basados en este sistema operativo. Ha dejado de ser fruto de cábalas y especulaciones, para convertirse en una realidad y ocupar una gran cuota de mercado, superando incluso, según las últimas estadísticas de ventas, al exitoso iPhone de Apple. De la mano de HTC, que apostó por él desde el principio y ha colaborado con Google en su lanzamiento y expansión (fabricando tanto los *Dev Phones* como el recién comercializado Nexus One), el resto de fabricantes se han subido al tren de Android y han desarrollado terminales con el nuevo sistema operativo. Como noticia de última hora en la fecha de redacción de este anexo, comentar que con la versión 2.2 (*Froyo*) que se lanzará próximamente (ya probada en un NexusOne), se espera un aumento del rendimiento del 450%, lo que situaría a este tipo de dispositivos a la cabeza del mercado en cuanto a prestaciones se refiere [23]. En el gráfico de la Figura B.2, puede verse la distribución de terminales comercializados según su versión de Android.

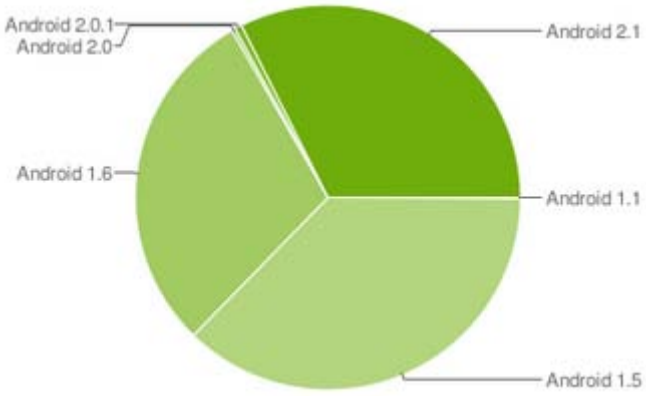


Figura B.2 – Distribución de las versiones de Android

En la fase inicial de este PFC, se tuvo que realizar un pequeño estudio para decidir qué dispositivos eran más apropiados para implementar el proyecto. Se analizaron por tanto los terminales existentes, que por aquel entonces, no eran muchos, como puede verse en la siguiente tabla comparativa (ver Figura B.3) [24].

Distribuidor	T-Mobile	– (GSM / HSDPA)	– (GSM / EDGE)	Sprint	T-Mobile	T-Mobile
Fabricante	Motorola	Samsung	HTC	HTC	HTC	HTC
Precio	–	–	–	$179.99	$99.99	$149.99
Presentado	–	July 2009	–	Oct 11, 2009	Aug 5, 2009	Oct 22, 2008
Teclado	Deslizante	Virtual	Virtual	Virtual	Virtual	Deslizante
Android Interfaz	MOTOBLUR	Standard	Sense	Sense	Standard	Standard
Procesador	528MHz MSM7201A	528MHz ARM11	528MHz MSM7225	528MHz MSM7201A	528MHz MSM7201A	528MHz MSM7201A
Pantalla	3.1-inch, 320 x 480	3.2-inch, 320 x 480	2.8-inch, 240 x 320	3.2-inch, 320 x 480	3.2-inch, 320 x 480	3.2-inch, 480 x 320
Auriculares	3.5mm	3.5mm	3.5mm	3.5mm	ExtUSB	ExtUSB
Tipo pantalla	Capacitiva	Capacitiva	Resistiva	Capacitiva	Capacitiva	Capacitiva
Camara	5MP con AF	5MP con Flash	3.2MP	5MP con AF	3.2MP con AF	3.2MP con AF
Bluetooth	2.0	2.1	2.0	2.0	2.0	2.0
Exchange	ActiveSync	ActiveSync	–	ActiveSync	Depends on version	Depends on version
Almacenam.	256MB, microSD	8GB, microSD	512MB, microSD	512MB, microSD	512MB, microSD	256MB, microSD
Bateria	1400mAh	1500mAh	1100mAh	1500 mAh	1340 mAh	1150 mAh
Peso	163g	114g	113g	135g	116g	158g

Figura B.3 – Comparativa terminales comerciales Android

Además de estos terminales comerciales, Google ofrecía dos dispositivos para desarrolladores, el Google Dev Phone 1 ó HTC G1 (equivalente al HTC Dream) y el Dev Phone 2 ó HTC G2 (equivalente al HTC Magic). Tras analizar detalladamente las diferentes posibilidades, el HTC Hero convencía por su potente *hardware* y altas prestaciones, sin bien los *Dev Phones* eran equipos proporcionados especialmente para el desarrollo, por lo que finalmente se optó

por adquirir dos terminales HTC G2 a través del Android Market. El motivo principal de tomar esta decisión fue que, en los terminales comerciales, era muy posible que diversos aspectos del teléfono vinieran bloqueados por el fabricante, como accesos al *hardware* o algunas partes del API de desarrollo.

B.4 Desarrollo de aplicaciones

Para el desarrollo de aplicaciones de Android [25], Google proporciona un *plugin* para Eclipse que facilita el trabajo, ofreciendo herramientas como emuladores, acceso al sistema de archivos, gestión de hilos en ejecución, consola, etc. Por lo tanto, para comenzar a trabajar, lo primero que debe hacerse es descargar Elipse, un IDE gratuito que puede obtenerse en la sección *downloads* del sitio oficial: http://www.eclipse.org. El programa no necesita instalación, basta con ejecutar el fichero eclipse.exe una vez descomprimido el archivo .zip descargado.

El siguiente paso será instalar el *plugin* de Android comentado. Para ello, hay que ir a la pestaña "Help > Install new software" (ver Figura B.4), y añadir la dirección web donde Eclipse deberá buscar y descargar el ADT (*Android Development Tool*), que puede encontrarse en la página para desarrolladores de Android, junto con los pasos detallados de la configuración: http://developer.android.com.

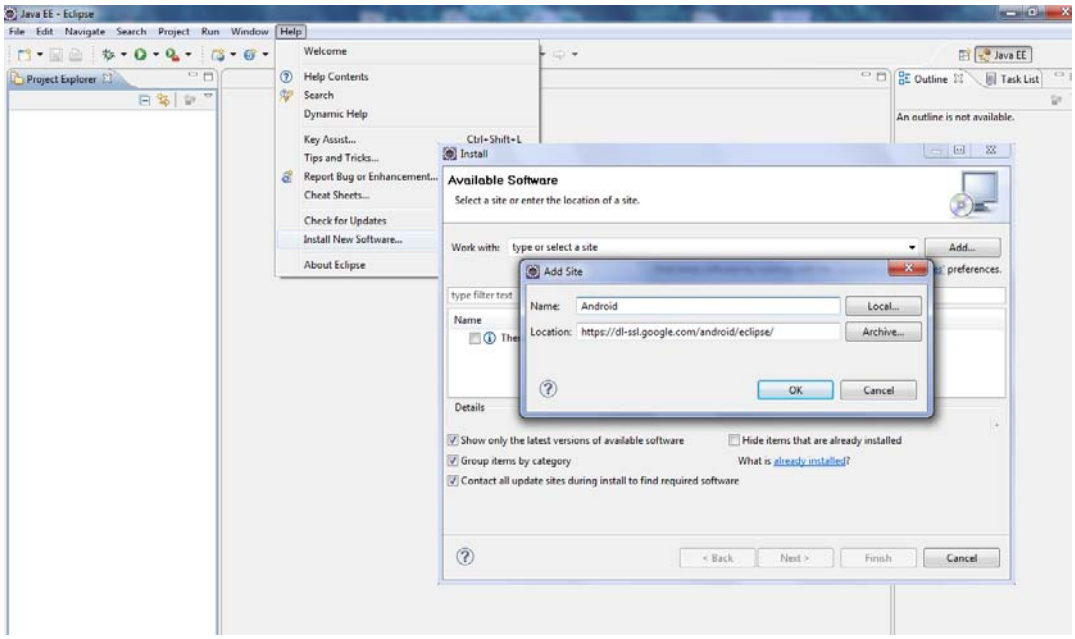


Figura B.4 – Instalación del plugin ADT en Eclipse

Por último, hay que descargar el SDK de desarrollo de Android, lo cual puede hacerse, al igual que en el caso anterior, desde la página oficial de Android. Antes de empezar a trabajar, se deberá asociar el *plugin* ya instalado con el SDK. Esto se consigue en la pestaña "Android" de la ventana "Window > Preferences", indicando la ubicación donde se encuentra la carpeta que contiene el SDK descargado (ver Figura B.5).

Desde este momento ya se puede comenzar a desarrollar aplicaciones para Android, pudiendo crear un nuevo proyecto, o cargar uno existente. Para este último caso, si se quisieran cargar los proyectos correspondientes al agente y al manager X73 realizados en este PFC, se debería ir a la pestaña "File > Import > Existing Projects into Workspace", eligiendo después la ubicación de trabajo (*workspace*) donde previamente se habrán copiado las carpetas correspondientes a los proyectos en cuestión.

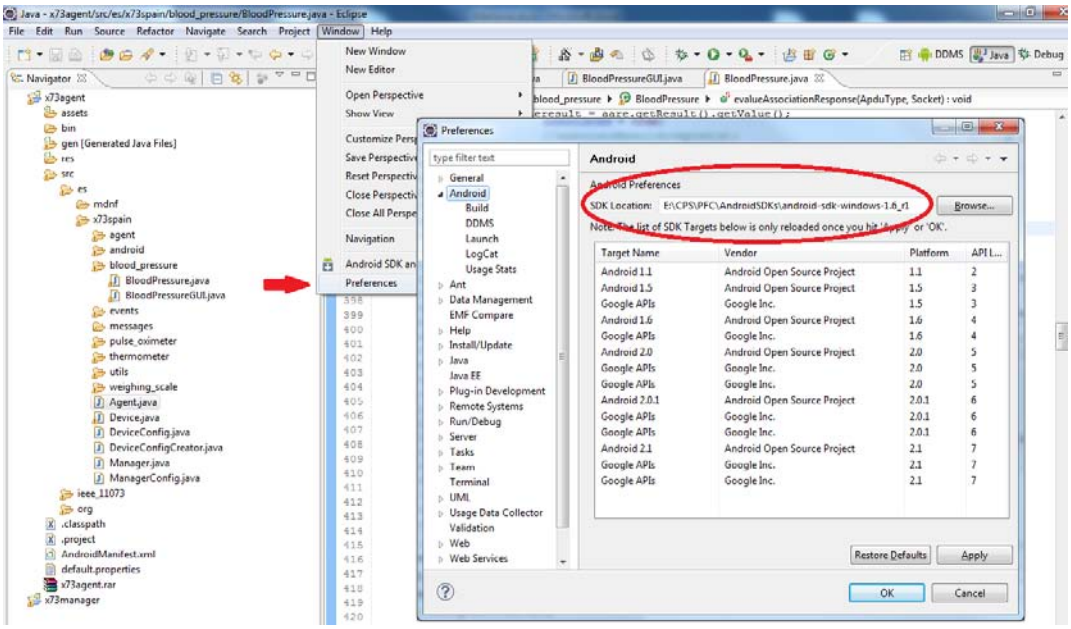


Figura B.5 – Configuración del plugin y SDK de Android en Eclipse

Otro aspecto importante es el de crear y configurar unidades virtuales (emuladores) para comprobar los programas en funcionamiento. Esto se consigue desde la ventana "Window > Android SDK and AVD Manager". Una vez en ella, en la pestaña "Virtual Devices", aparecen los emuladores de que se dispone (ninguno si es la primera que se utiliza). Para crear uno nuevo, basta con pulsar en "New" y especificar el nombre y la versión de Android con la que se quiere trabajar (ver Figura B.6).

Una vez creado, ya se puede ejecutar la aplicación deseada, pinchando con el botón derecho del ratón en el proyecto, después en la opción “Rus As > Android Application”, y finalmente eligiendo el emulador, o dispositivo real si lo hubiera, que se quiere utilizar.

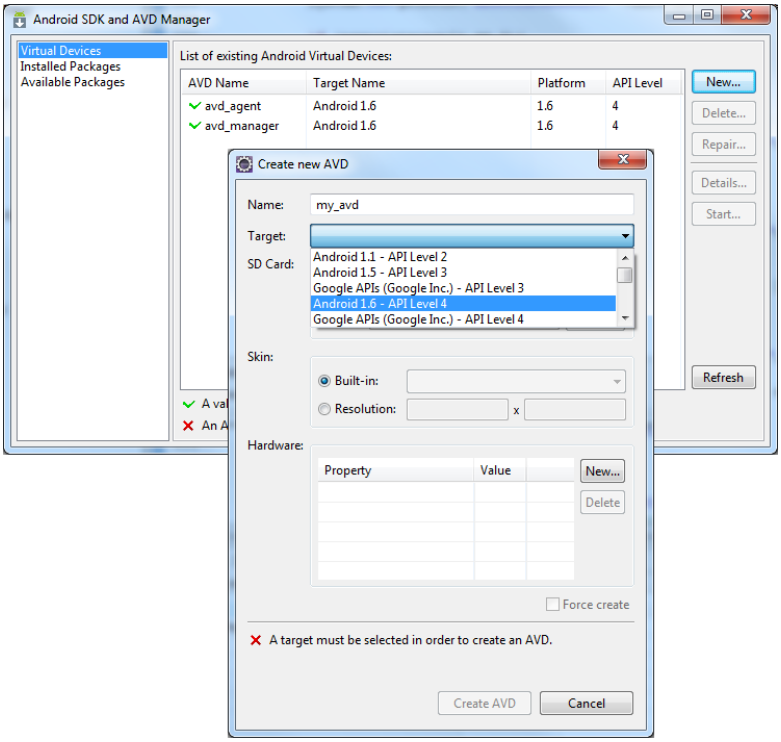


Figura B.6 – Creación de un emulador de Android con Eclipse

Una utilidad importante es la herramienta DDMS que viene incluida en el *plugin*. Contiene varias funciones, pero las más importantes son la consola “Log” y el “File Explorer”. Con esta consola se pueden visualizar mensajes incluidos en el código, lo que permite ver las funciones por las que pasa el programa, comprobar errores y ver que todo funciona correctamente. Se visualizan los mensajes tanto del propio sistema de *logs* que proporciona la pila de Android, como de la salida estándar “System.out” o la de errores “System.err”. Por otro lado, la pestaña “File Explorer” es muy útil cuando se trabaja con archivos, como es el caso de este proyecto, donde se crean ficheros XML en los que se almacenan los datos médicos. Esta herramienta permite explorar en el sistema de archivos del terminal (o emulador), algo muy importante ya que el emulador no incluye por defecto un explorador de archivos.

Anexo C – Definición de medidas

En este Anexo se definirán las medidas con las que se trabaja en el proyecto, especificando cómo se deben tomar para que el dato médico sea válido y representativo de la salud del paciente, ya que una medida tomada de forma errónea, derivaría en diagnósticos incorrectos, lo que podría dar lugar a pasar por alto ciertos problemas de salud o enfermedades.

C.1 Temperatura

La **temperatura corporal** [26] se define como el grado de calor que tiene el cuerpo humano, y que se consigue con la diferencia entre el calor que se produce en nuestro organismo a través del metabolismo, y el enfriamiento que producido por agentes externos. El centro regulador de la temperatura en el organismo es el hipotálamo.

El ser humano es un animal homeotermo que en condiciones fisiológicas normales mantiene una temperatura corporal constante y dentro de unos límites muy estrechos, entre 36 y 37 ºC aproximadamente, a pesar de las amplias oscilaciones de la temperatura ambiental. Si bien puede experimentar cambios en relación al ejercicio, al ciclo menstrual, a los patrones de sueño, a la temperatura del medio ambiente, etc. Cuando la temperatura corporal se encuentra fuera el rango mencionado, se puede distinguir entre los siguientes casos:

- **Hipotermia**: Cuando la temperatura es inferior a los 36 ºC.
- **Febrícula**: Si ésta se encuentra entre 37 y 38 ºC
- **Hipertermia** o **fiebre**: Cuando la temperatura es superior a 38 ºC.

La temperatura se puede medir en diferentes lugares del cuerpo humano, siendo los más típicos la axila, la boca y el recto.

- **Temperatura axilar**: Es la más cómoda y segura, aunque también la menos exacta, ya que es fuertemente influida por la temperatura externa. Se coloca el termómetro bajo la axila y se espera de 3 a 5 minutos para su lectura.
- **Temperatura oral** o **bucal**: Entre sus ventajas se encuentran el ser accesible y cómoda, además de bastante fiable. Se coloca en la boca el termómetro, bajo la lengua y se espera de 3 a 5 minutos para la lectura.

- **Temperatura rectal**: Es la más exacta de las tres, aunque a la vez es la más incómoda. Está indicada en los niños menores de 6 años y en los enfermos inconscientes o confusos.

Existen varios tipos de termómetros, dependiendo de su funcionamiento y de las formas de uso, pero los más comunes son:

- **Termómetro de mercurio**: Termómetro de cristal, con cuerpo tubular y de sección triangular, con mercurio como material indicador. Tiene un rango de 35 a 42 ºC. Los hay axilar (punta larga de mercurio) y rectal (punta corta de mercurio). El tiempo de medida es de 3 a 5 minutos.
- **Termómetro digital**: termómetro electrónico digital, que trabaja con una pila alcalina y posee pantalla de lectura. Puede ser utilizado para tomar temperatura oral, axilar o rectal, en un tiempo de 60 segundos.

C.2 Tensión

La **presión arterial** (PA) o **tensión arterial** (TA) es la presión que ejerce la sangre contra la pared de las arterias [27]. Esta presión es imprescindible para que circule la sangre por los vasos sanguíneos y aporte el oxígeno y los nutrientes a todos los órganos del cuerpo para que puedan funcionar. Es un caso particular de la presión sanguínea, si bien en numerosas ocasiones se utilizan ambos términos referidos a lo mismo. La presión arterial tiene dos componentes:

- **Presión arterial sistólica:** corresponde al valor máximo de la tensión arterial en sístole (cuando el corazón se contrae).
- **Presión arterial diastólica:** corresponde al valor mínimo de la tensión arterial cuando el corazón está en diástole o entre latidos cardíacos.

La presión arterial, como medida de presión que se trata, se mide normalmente en miligramos de mercurio (mmHg), y sus valores típicos suelen estar entre 90/60 y 120/80 mmHg. (sistólica/diastólica). Valores por encima de 130/90 mmHg indicarán que el paciente sufre **hipertensión**, mientras que por debajo de 90/60 mmHg se tratará de **hipotensión**.

Existen varios tipos de aparatos para medir la tensión: el **tensiómetro de mercurio**, **tensiómetro de aire**, **medidor electrónico** y el **holter**. Los tensiómetros de mercurio y de aire cuentan también con un estetoscopio, en ambos casos se coloca una banda alrededor del

brazo, unos centímetros por encima del codo y entre está y el brazo se coloca el estetoscopio. Al inflarse la banda, ejerce presión deteniendo el flujo normal de la sangre, luego se deja salir el aire y se mide la presión durante el latido del corazón (sistólica) y mientras el corazón descansa (diastólica).

Sin embargo, tanto el tensiómetro electrónico (usado a nivel de hogar) como el holter (a nivel hospitalario) son mucho más precisos que los anteriores, y son los más utilizados.

C.3 Pulso

El **pulso** [27], es la expansión rítmica de una arteria, producida por el paso de la sangre bombeada por el corazón. Se controla para determinar el funcionamiento del corazón, siendo además un método rápido y sencillo para valorar el estado de un lesionado.

Se obtiene por lo general en partes del cuerpo donde las arterias se encuentran más próximas a la piel, como en las muñecas o el cuello, palpando manualmente con los dedos índice y corazón; no se puede tomar con el dedo pulgar ya que este tiene pulso propio. Una forma alternativa de encontrar el pulso es oír el latido del corazón, lo que suele hacerse con un estetoscopio.

Un pulso normal para un adulto sano en descanso oscila entre 60 y 100 pulsaciones por minuto. Durante el sueño puede caer hasta las 40 pulsaciones y durante el ejercicio intenso puede subir hasta las 200 pulsaciones. El factor más influyente en el pulso es la edad del individuo y, en función de ésta, se establecen unos rangos orientativos de pulsaciones por minuto:

- Bebés: 130 – 140 ppm
- Niños: 80 – 100 ppm
- Adultos: 70 – 80 ppm
- Ancianos: 60 o menos ppm

Aparte de su velocidad, el pulso tiene otras cualidades que reflejan el estado del sistema cardiovascular, tales como su ritmo, amplitud y forma de la onda de pulso. Ciertas enfermedades provocan cambios característicos en estas cualidades, por lo que se trata de una medida importante e indicativa de la salud del paciente.

Una frecuencia cardiaca en reposo consistentemente alta (taquicardia) pude ser indicio de un problema y debe consultarse, al igual que frecuencias por debajo de los valores normales (bradicardia).

C.4 Nivel de oxígeno

El **nivel de oxígeno en sangre** [28], o **saturación** (**SpO2**), describe el grado de capacidad de transporte de oxígeno de la hemoglobina en los vasos sanguíneos. La oximetría de pulso o pulsioximetría es por tanto la medición, no invasiva, de este nivel de oxígeno transportado por los glóbulos rojos.

El color de la sangre varía dependiendo de lo saturada de oxígeno que se encuentre, debido a las propiedades ópticas de la molécula de hemoglobina. Cuando esta molécula libera oxígeno pierde su color rosado, adquiriendo un tono más azulado y deja pasar menos la luz roja.

Así pues el pulsioxímetro determina la saturación de oxígeno midiendo el "grado" de azules de la sangre arterial y expresa esta "azulez" en términos de saturación. Dado que la absorción de luz de los tejidos y de la sangre venosa son constantes, cualquier cambio en la absorción de la luz entre un tiempo dado y uno posterior se deben exclusivamente a la sangre arterial. Los pulsioxímetros miden pues la relación, en un intervalo de tiempo, entre las diferencias de absorción de las luces rojas e infrarroja. Esta relación se vincula directamente con la saturación de oxihemoglobina.

Estos aparatos funcionan con una pinza que tiene un productor de luz, la cual se refleja en la piel del dedo; este sensor mide la cantidad de luz absorbida por la oxihemoglobina circulante en el paciente. Normalmente, los pulsioxímetros proporcionan tres medidas: el índice de saturación de oxígeno, la frecuencia cardiaca, y la curva del pulso.

La saturación de oxígeno debe ser mayor del 95%. Valores aumentados serían síntoma de **hiperventilación** o **ansiedad**, mientras que valores menores indicarían enfermedades pulmonares crónicas, descompensación o crisis de asma, o enfermedades cardíacas.

C.5 Peso

El **peso**, según la OMS [29], es la medida de valoración nutricional más empleada, cuyo concepto se remonta a la antigua Grecia hace más de 2000 años. Por suerte, las balanzas que permiten su medición han evolucionado y hoy en día no representa ningún obstáculo el llevarlo a cabo, incluso en personas enfermas cuya movilidad sea dificultosa.

El peso, no obstante, está en función del tipo morfológico y del esqueleto del individuo, por ello en ocasiones es preferible tomar otras medidas más representativas, como puede ser el **Índice de Masa Corporal (IMC)**. Este índice es una medición indirecta de la composición corporal y tiene en cuenta tanto el peso como la estatura:

$$IMC = \frac{peso}{2 * altura}$$

Según el valor del IMC, es posible clasificar a los individuos en varias categorías (Tabla C.1). La columna "Riesgo" indica el riesgo de sufrir otras enfermedades, a parte de la obesidad, como pueden ser: anorexia, bulimia, trastornos cardiovasculares, muerte prematura, hipertensión, etc.

Tabla C.1 – Clasificación Indice de Masa Corporal

Clasificación	IMC (Kg/m^2)	Riesgo
Peso bajo	< 18.5	Alto
Normal	18.5 – 24.9	Promedio
Sobrepeso	25.0 – 29.9	Aumentando
Obesidad	30.0 – 39.9	Severo
Obesidad extrema	> 39.9	Muy severo

Por lo tanto, visto que el peso es un elemento determinante en la contracción de enfermedades, es importante mantenerlo bajo control, siguiendo las siguientes recomendaciones:

- Llevar una dieta sana y equilibrada.
- Compaginar la dieta con actividad física; el ejercicio aeróbico ayuda a incrementar el tejido muscular y a quemar calorías.
- Evitar el alcohol o beber con moderación.
- Recurrir a especialistas si no se consiguen cambiar los hábitos de alimentación.